INFORMÁTICA EDUCATIVA

Hasier MORRÁS ARANOA

INICIACIÓN A LA PROGRAMACIÓN INFORMÁTICA EN EDUCACIÓN PRIMARIA CON SCRATCH

TFG/*GBL* 2014

upna Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Facultad de Ciencias Humanas y Sociales
Giza eta Gizarte Zientzien Fakultatea

Grado en Maestro de Educación Primaria /
Lehen Hezkuntzako Irakasleen Gradua

Grado en Maestro en Educación Primaria

Lehen Hezkuntzako Irakasleen Gradua

Trabajo Fin de Grado
Gradu Bukaerako Lana

INICIACIÓN A LA PROGRAMACIÓN INFORMÁTICA EN EDUCACIÓN PRIMARIA CON SCRATCH

Hasier MORRÁS ARANOA

FACULTAD DE CIENCIAS HUMANAS Y SOCIALES
GIZA ETA GIZARTE ZIENTZIEN FAKULTATEA

UNIVERSIDAD PÚBLICA DE NAVARRA
NAFARROAKO UNIBERTSITATE PUBLIKOA

Estudiante / Ikaslea
Hasier MORRÁS ARANOA

Título / Izenburua
Iniciación a la programación informática en Educación Primaria con Scratch

Grado / Gradua
Grado en Maestro en Educación Primaria / Lehen Hezkuntzako Irakasleen Gradua

Centro / Ikastegia
Facultad de Ciencias Humanas y Sociales / Giza eta Gizarte Zientzien Fakultatea
Universidad Pública de Navarra / Nafarroako Unibertsitate Publikoa

Director-a / Zuzendaria
Alfredo PINA CALAFI

Departamento / Saila
Ingeniería Matemática e Informática / Matematika eta Informatika Ingeniaritza

Curso académico / Ikasturte akademikoa
2013/2014

Semestre / Seihilekoa
Primavera / Udaberria

Preámbulo

El Real Decreto 1393/2007, de 29 de octubre, modificado por el Real Decreto 861/2010, establece en el Capítulo III, dedicado a las enseñanzas oficiales de Grado, que "estas enseñanzas concluirán con la elaboración y defensa de un Trabajo Fin de Grado […] El Trabajo Fin de Grado tendrá entre 6 y 30 créditos, deberá realizarse en la fase final del plan de estudios y estar orientado a la evaluación de competencias asociadas al título".

El Grado en Maestro en Educación Primaria por la Universidad Pública de Navarra tiene una extensión de 12 ECTS, según la memoria del título verificada por la ANECA. El título está regido por la *Orden ECI/3857/2007, de 27 de diciembre, por la que se establecen los requisitos para la verificación de los títulos universitarios oficiales que habiliten para el ejercicio de la profesión de Maestro en Educación Primaria*; con la aplicación, con carácter subsidiario, del reglamento de Trabajos Fin de Grado, aprobado por el Consejo de Gobierno de la Universidad el 12 de marzo de 2013.

Todos los planes de estudios de Maestro en Educación Primaria se estructuran, según la Orden ECI/3857/2007, en tres grandes módulos: uno, *de formación básica*, donde se desarrollan los contenidos socio-psico-pedagógicos; otro, *didáctico y disciplinar*, que recoge los contenidos de las disciplinares y su didáctica; y, por último, *Practicum*, donde se describen las competencias que tendrán que adquirir los estudiantes del Grado en las prácticas escolares. En este último módulo, se enmarca el Trabajo Fin de Grado, que debe reflejar la formación adquirida a lo largo de todas las enseñanzas. Finalmente, dado que la Orden ECI/3857/2007 no concreta la distribución de los 240 ECTS necesarios para la obtención del Grado, las universidades tienen la facultad de determinar un número de créditos, estableciendo, en general, asignaturas de carácter optativo.

Así, en cumplimiento de la Orden ECI/3857/2007, es requisito necesario que en el Trabajo Fin de Grado el estudiante demuestre competencias relativas a los módulos de formación básica, didáctico-disciplinar y practicum, exigidas para todos los títulos universitarios oficiales que habiliten para el ejercicio de la profesión de Maestro en Educación Primaria.

En este trabajo, el módulo *de formación básica* se concreta en las aportaciones a la pedagogía y al desarrollo cognitivo de los autores estudiados, aportaciones que han resultado imprescindibles al diseñar nuestra propuesta educativa. Estas aportaciones se explicitarán en las dos primeras secciones del trabajo.

Además, la asignatura *Habilidades comunicativas y TIC* cobrará mayor relevancia por el carácter informático del trabajo.

El módulo *didáctico y disciplinar* puede verse reflejado en la planificación de las actividades propuestas, ya que dicho módulo abarca los aspectos relacionados con el proceso de enseñanza-aprendizaje.

Las asignaturas de *Matemáticas y su didáctica I* y *II* van a estar especialmente presentes dadas las exigencias de conocimientos matemáticos incluidas en la programación informática, aunque a su vez el proyecto Scratch es aplicable a múltiples asignaturas.

Asimismo, el módulo *practicum* nos ha permitido observar y vivir la realidad del aula y el funcionamiento de un centro escolar, y todo ello nos ha ayudado a adaptar nuestra propuesta educativa a situaciones lo más reales posibles.

Resumen

Este trabajo propone introducir contenidos de programación informática en Educación Primaria a través del proyecto Scratch y reflexionar sobre sus aplicaciones didácticas y sobre su relación con el currículo oficial de Navarra.

El fin subyacente en esta propuesta es fomentar el uso de las TIC para producir contenidos digitales, más allá del nivel de usuario competente, es decir, pasar de usuario a productor, utilizando las TIC como herramienta de creación, y se analizarán las implicaciones que ello conlleva. También se expondrán las contribuciones de la programación informática a la educación y al desarrollo de la competencia digital.

Para comprobar la viabilidad de la propuesta se planificará un estudio de caso en el que se introducirá al alumnado de Educación Primaria en el uso de Scratch y se evaluarán los resultados obtenidos.

Palabras clave: Scratch; Educación Primaria; programación informática; TIC; Navarra

Abstract

This work proposes to introduce computer programming contents in Primary Education through the Scratch project and to reflect on its educational uses and its relationship with the official curriculum of Navarre.

The underlying aim of this proposal is to foster the use of ICT to produce digital contents, going beyond the competent user level, that is to move from user to producer, using the ICT as a creation tool, and the implications involving this matter will be analysed. The contributions of computer programming in education and in the development of the digital competence will also be presented.

In order to check the viability of the proposal a case study where Primary Education pupils are introduced into the use of Scratch will be planned and the obtained results will be evaluated.

Keywords: Scratch; Primary Education; computer programming; ICT; Navarre

Laburpena

Lan honek Scratch proiektuaren bidez Lehen Hezkuntzan programazio informatikoko edukiak sartzea proposatzen du bai eta proiektu honen aplikazio didaktikoei eta Nafarroako curriculum ofizialarekiko duen erlazioari buruz gogoeta egitea ere.

Proposamen honen azpiko asmoa IKTen erabilera sustatzea da eduki digitalak sortzeko, erabiltzaile adituaren mailatik haratago, hau da, erabiltzailetik ekoizlera pasatzea, IKT sortze tresna bezala erabiliz, eta honek dakartzan inplikazioak ere aztertuko dira. Programazio informatikoak hezkuntzan eta gaitasun digitalean dituen ekarpenak ere azalduko dira.

Proposamenaren bideragarritasuna egiaztatzeko kasu azterketa baten plangintza egingo da non Lehen Hezkuntzako ikaslegoa Scratch-en erabileran hasiko da eta lortutako emaitzak balioztatuko diren.

Hitz gakoak: Scratch; Lehen Hezkuntza; programazio informatikoa; IKT; Nafarroa

Índice

INTRODUCCIÓN

Los cambios vertiginosos que está viviendo nuestra sociedad actual a raíz de los avances e innovaciones en las Tecnologías de la Información y la Comunicación[1] inciden directamente en todos los niveles de la educación. Es por ello labor de los educadores y educadoras formarnos en TIC e investigar nuevas formas de integrarlas en el aula.

En el caso de Educación Primaria, la asignatura *Habilidades comunicativas y TIC* del Grado en Maestro nos deja claro que los maestros y maestras debemos ser usuarios competentes de las TIC y fomentar su uso en el aula.

Sin embargo, si indagamos más profundamente en las TIC veremos que en su base se encuentra la programación informática. Es el primer paso, el guión por el que se regirán las diferentes aplicaciones informáticas o incluso los aparatos electrónicos. Es por ello que en una sociedad cada vez más digitalizada poseer conocimientos de programación informática puede resultar de gran provecho, además de las aplicaciones didácticas que pueda tener.

Siendo conscientes de la dificultad de la programación informática, proponemos un paso previo, una iniciación a este campo a través de un proyecto realizado por el área pedagógica del Masacchussets Institute of Tecnhology llamado Scratch, que consiste en una plataforma que permite programar de manera simplificada y más visual, pensada especialmente para el alumnado de Educación Primaria y Secundaria.

Comenzaremos por dejar claro qué es la programación informática y qué usos tiene, y veremos una breve evolución histórica de la misma para ver en qué punto estamos hoy en día. Veremos también las adaptaciones que se han creado además de Scratch.

Seguidamente haremos un esbozo de la situación actual de las TIC en los proyectos educativos a nivel europeo, estatal y regional, así como un escaneo de las escuelas de Navarra a este respecto.

[1] A partir de este momento, el término "Tecnologías de la Información y la Comunicación" aparecerá como TIC.

En el marco teórico relacionaremos el tema escogido en este trabajo con teorías del aprendizaje ampliamente aceptadas por la comunidad pedagógica, y haremos lo mismo con las corrientes de pensamiento actuales. Analizaremos las implicaciones docentes que pueda tener y relacionaremos una vez más la programación informática con el currículo oficial de Educación Primaria de Navarra.

Tras el marco teórico propondremos la realización de un estudio de caso para el que describiremos el material necesario y la metodología utilizada. El objetivo de este estudio de caso será comprobar si nuestra propuesta de utilizar el proyecto Scratch en diferentes cursos de Educación Primaria es viable o no, o en qué cursos es adecuado y en cuáles no.

Para realizar la comprobación mostraremos un modelo de rúbrica de evaluación del estudio de caso con el cual poder fundamentar la adecuación de la propuesta. Asimismo, se discutirán los diferentes aspectos que puedan variar los resultados.

Por último expondremos nuestras conclusiones y dejaremos abiertas ciertas cuestiones que por la extensión del trabajo puedan quedar sin cubrir.

Nota:

Las referencias a personas o colectivos figuran en el presente trabajo en género masculino como género gramatical no marcado. Cuando proceda, será válida la cita de los preceptos correspondientes en género femenino.

1 ANTECEDENTES, OBJETIVOS Y CUESTIONES

1.1 La programación informática

1.1.1 Concepto

La programación informática es un proceso en el cual el programador escribe instrucciones en un lenguaje de programación para que un ordenador u otro aparato electrónico ejecute esas órdenes[2].

Esta definición tiene innumerables aplicaciones. Como ejemplo podríamos mencionar que la programación informática se encuentra en:

- Programas informáticos o software (incluyendo Sistemas Operativos).
- Páginas web.
- Videojuegos.
- Robótica (los robots necesitan seguir órdenes que se hayan programado previamente).
- Maquinaria industrial.
- Electrodomésticos.
- Etc.

El programador informático es el que escribe las instrucciones del programa, aunque hoy en día sus funciones en el mercado laboral se han expandido a otras áreas como la planificación y gestión de sistemas dentro de una empresa, la seguridad informática o la asistencia técnica.

El lenguaje de programación es un conjunto de símbolos y reglas semánticas y sintácticas que crean instrucciones legibles para el ordenador u otros aparatos electrónicos. Los programadores utilizan lenguajes de alto nivel para programar, y dichos lenguajes se traducen a lenguajes de bajo nivel que son los que las máquinas

[2] Información obtenida de la página web
(http://www.computerhope.com/jargon/p/progming.htm, 24/04/2014)

reconocen directamente.[3] Los lenguajes de programación están escritos en inglés, aunque como veremos más adelante hay adaptaciones en diferentes idiomas.

1.1.2 Historia y evolución

Balagurusamy (2010) realizó una breve descripción histórica y evolutiva de la programación informática en la que nos hemos basado para elaborar la siguiente tabla:

Tabla 1. Historia y evolución de la programación informática

Periodo	**Acontecimiento**
1843	Ada Lovelace (hija de Lord Byron) está considerada la primera programadora informática al haber escrito unas instrucciones para la "máquina analítica" de Charles Babbage en las que permitía traducir órdenes del usuario a un formato binario que la máquina podía comprender.
1946	Konrad Zuse crea Plankalkül, considerado el primer lenguaje de programación completo.
1940-1950	En esta década se utilizan "lenguajes de máquina" binarios (utilizando solamente el 1 y el 0) que son lenguajes de bajo nivel y particulares de cada máquina.
1950-1960	En esta década se utiliza el "lenguaje ensamblador" que, aunque sigue siendo un lenguaje de bajo nivel, traduce los códigos de la máquina en palabras o símbolos legibles (llamados mnemónicos) para el programador.
1952	Grace Hopper, considerado el primer programador centrado en crear un lenguaje de alto nivel, crea A-0.
1957	John Backus, junto con su equipo de IBM, crea el popular FORTRAN, que se convertirá en el primer lenguaje de alto nivel utilizado ampliamente.
1964	John G. Kemeny y Thomos E. Kurtz crean el primer lenguaje de alto nivel pensado para principiantes llamado BASIC.
1970-1980	Esta década está considerada como la época dorada de la programación informática, ya que se crean varios lenguajes de muy alto nivel que incluso hoy en día se siguen utilizando, como Pascal, SQL o C.
1990-presente	La evolución de Internet hace que aparezcan lenguajes de programación centrados en la red, como Java, HTML o PHP. Hoy en día la tendencia es que las grandes compañías de TIC como Google, Mozilla o Facebook creen sus propios lenguajes de programación, como Dart, Rust o Hack, respectivamente.

[3] Información obtenida de la página web (http://www.techterms.com/definition/programming_language, 26/04/2014)

1.1.3 Adaptaciones: Logo y Scratch

A lo largo de la breve pero intensa historia de los lenguajes de programación se han creado diferentes adaptaciones de éstas con fines educativos y simplificadores como Greenfoot, Etoys, Alice, Logo o Scratch. Esto ha ocurrido debido a la dificultad y la falta de estética (sólo texto) de los lenguajes, como hemos mencionado anteriormente y como podemos ver en la figura 1.

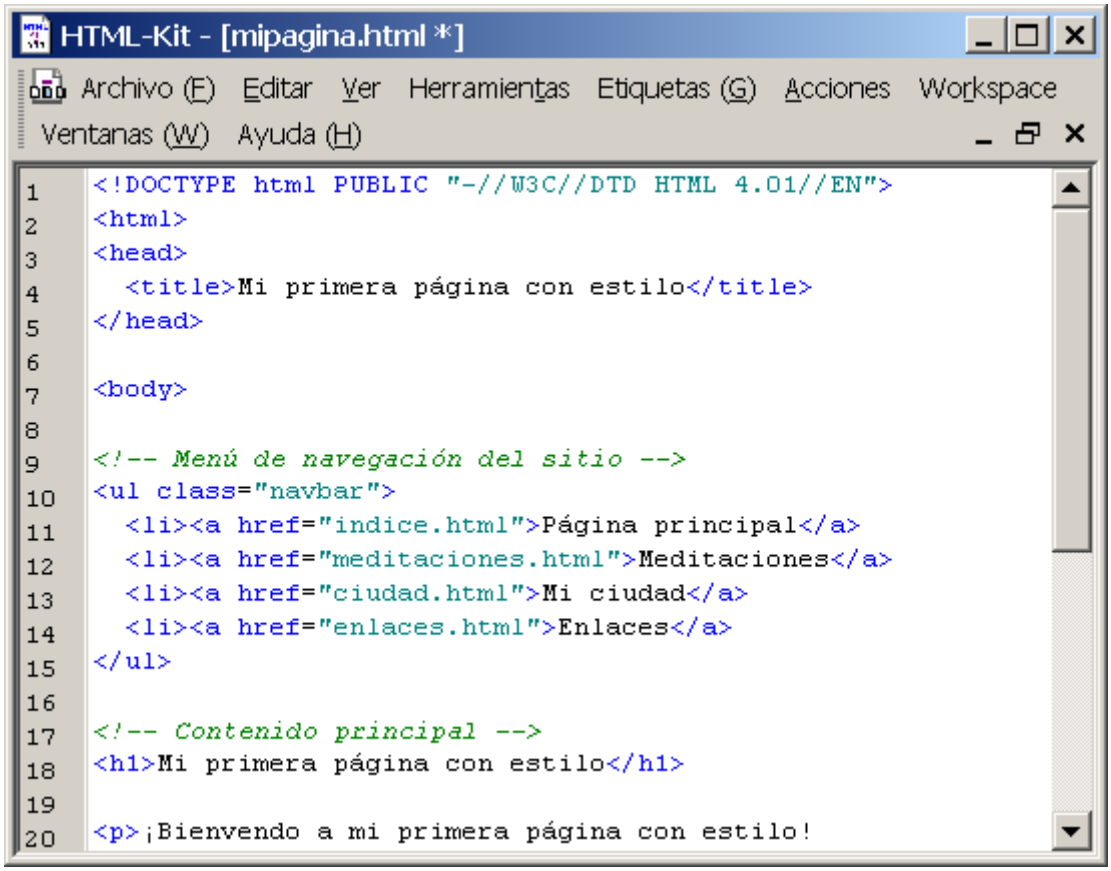


Figura 1. Editor de HTML. Obtenida de la página web (http://www.w3.org/Style/Examples/011/firstcss.es.html, 28/04/2014)

En este trabajo nos centraremos en dos de los proyectos que mayor acogida han tenido en el mundo escolar, que son Logo y Scratch, el primero con una larga trayectoria aunque menos atractivo y el segundo de reciente creación y altamente motivador.

En 1967 un grupo de investigadores del ámbito de la inteligencia artificial, entre los que se encontraba el también educador Seymour Papert, creó el lenguaje de programación educativo Logo (a su vez basado en otro lenguaje llamado Lisp) con el principal objetivo de aportar un nuevo método de enseñanza de las matemáticas.

Según Feurzeig, Papert, y Lawler (2011) la mayoría de métodos formales de la enseñanza de las matemáticas resultan tediosos, poco motivadores y poco relacionados con el pensamiento intuitivo del alumnado. Para superar dichos obstáculos proponen una adecuada instrucción en programación informática.

Viendo la figura 1 parece obvio que una adecuada instrucción en lenguajes de programación requiere de una iniciación utilizando un lenguaje adaptado, más sencillo y más visual, sobre todo en el alumnado más joven.

Es por ello que llevaron a cabo el proyecto Logo, que tuvo una gran aceptación en la comunidad educativa, aplicándose en numerosas escuelas. Coloquialmente se le llama "el lenguaje de la tortuga", y en la figura 2, un ejemplo de Logo, podemos ver el por qué (se trata de programar el movimiento de la tortuga indicando la dirección y la cantidad de movimiento para conseguir diferentes objetivos).

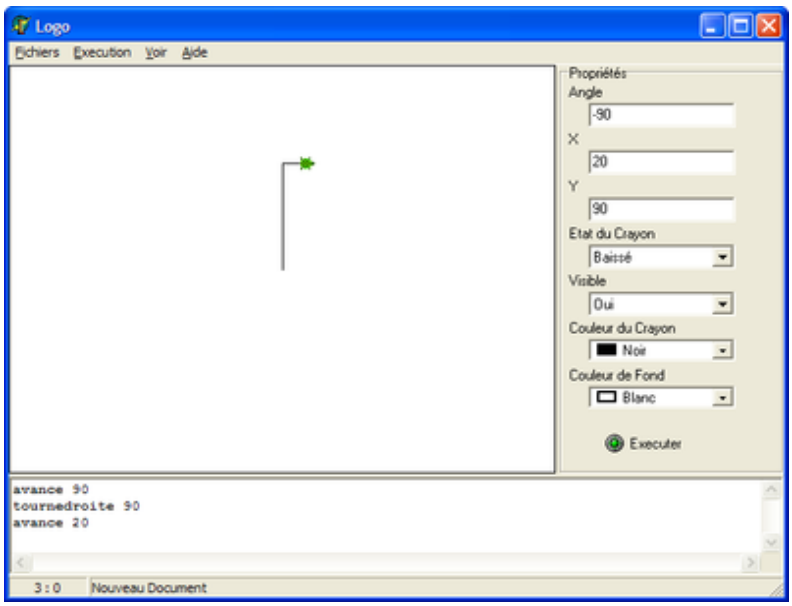


Figura 2. Editor de Logo. Obtenida de la página web (http://en.kioskea.net/download/download-18415-logo, 01/05/2014)

Logo ha influido enormemente en los lenguajes de programación educativos, y en uno de los lenguajes actuales que mayores éxitos está cosechando, Scratch, también podemos ver ésa influencia.

El proyecto Scratch surgió en 2003 del laboratorio de investigaciones interdisciplinares MIT Media Lab del Instituto Tecnológico de Massachusetts, más concretamente de la rama educativa Lifelong Kindergarten, aunque no fue hasta 2006 cuando se creó la página web del proyecto desde el cual cualquier usuario podía descargar el programa Scratch 1.0.

Sin embargo a partir de 2013 con la versión 2.0 no es necesario descargar ningún programa, ya que se puede editar con Scratch directamente desde su página web. Está disponible en varios idiomas, incluido el castellano, y todo esto ha propiciado que exista una gran comunidad de usuarios a nivel mundial compartiendo sus creaciones. Además, las versiones anteriores estaban escritas en el lenguaje Squeak (a su vez influenciado por Logo), pero la versión actual está escrita en un lenguaje de programación estandarizado llamado ActionScript, que se utiliza para crear animaciones y aplicaciones Flash. Esto supone que las creaciones hechas con Scratch sean fácilmente exportables a otras páginas web, ganando en compatibilidad.

El término Scratch viene de la técnica “scratching” que realizan los “disc jockeys” al remezclar canciones, y la idea es que similarmente los usuarios de Scratch puedan remezclar diferentes tipos de contenidos multimedia fácilmente[4]

El objetivo de Scratch, a diferencia de Logo y las matemáticas, no se centra en una sola materia sino que intenta ser una plataforma multidisciplinar donde se desarrollan diferentes capacidades del usuario.

Según los creadores, ayuda a los jóvenes a aprender a pensar creativamente, razonar sistemáticamente, y trabajar colaborativamente, lo que consideran habilidades

[4] Información obtenida de la página web
(http://newsoffice.mit.edu/2007/resnick-scratch, 01/05/2014)

esenciales para la vida en el siglo XXI, y uno de sus lemas es “aprender a programar, programar para aprender”[5].

Uno de los puntos fuertes de este lenguaje es que es muy visual, es decir, no es necesario usar el teclado para escribir, sino que con el puntero del ratón se pueden arrastrar “bloques” con comandos predefinidos. En la figura 3 podemos observar el entorno estéticamente agradable, accesible y sencillo del editor.

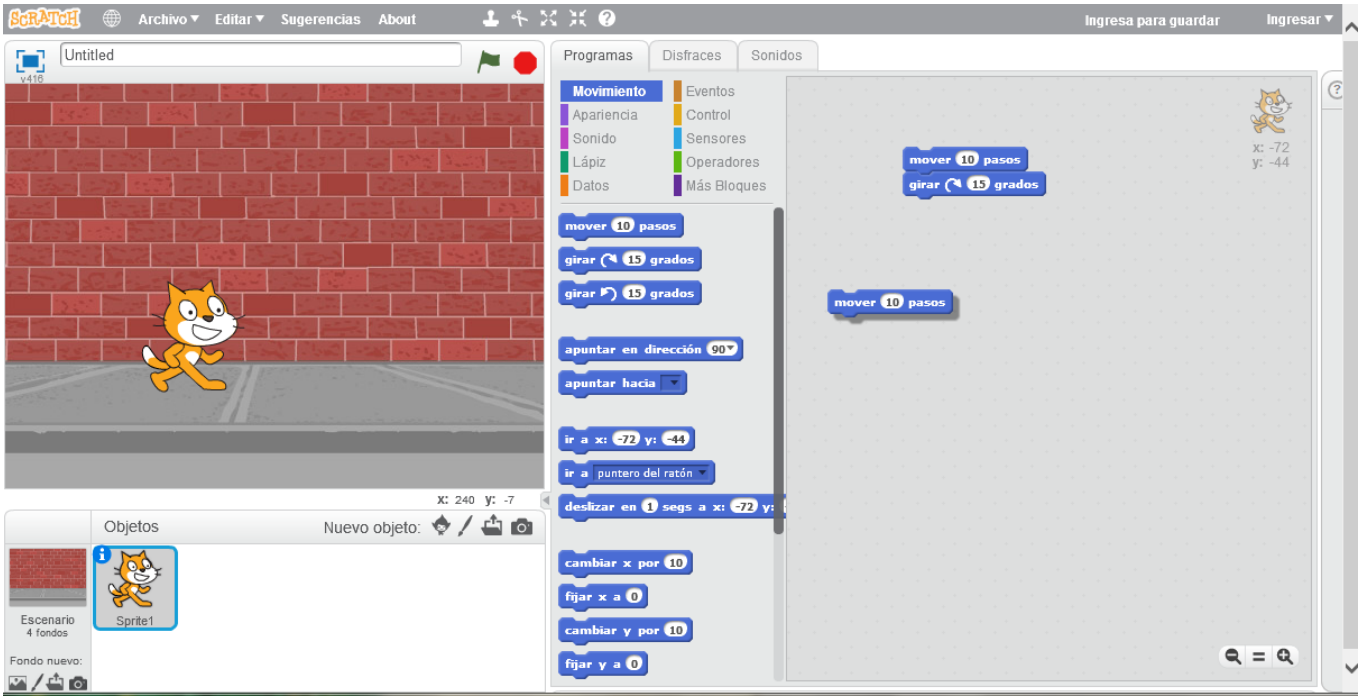


Figura 3. Editor Scratch. Obtenida de la página web (http://scratch.mit.edu/projects/editor/?tip_bar=getStarted, 01/05/2014)

Queda así claro que la forma más adecuada de introducir la programación informática en edades tempranas es a través de lenguajes de programación adaptados, y creemos que de entre ellos Scratch es el lenguaje que mejor puede cubrir nuestras demandas educativas.

[5] Información obtenida de la página web (http://scratch.mit.edu/about/, 01/05/2014)

1.2 Sociedad de la información

La agencia especializada Unión Internacional de Telecomunicaciones (UIT) de Naciones Unidas organizó la Cumbre Mundial sobre la sociedad de la información en dos fases, la primera en Ginebra en 2003 y la segunda en Túnez en 2005, y entre los puntos de discusión se encontraban cómo hacer frente a la llamada "brecha digital" o la creación del Foro de Gobernanza de Internet. Asimismo, se creó el Grupo de las Naciones Unidas sobre la Sociedad de la Información[6].

Es indudable que nuestra sociedad se está sumergiendo en una época donde el intercambio de información a través de las TIC ha cobrado gran relevancia en todos los aspectos, incluyendo la economía, la política, la cultura o la educación. Es lo que llamamos sociedad de la información, y este nuevo concepto podemos verlo reflejado en las diferentes leyes o proyectos educativos.

1.2.1 Planes europeos y estatales

Aunque dentro de la Unión Europea la educación sea responsabilidad de cada estado miembro, desde las instituciones europeas aportan apoyo, coordinación o iniciativas complementarias. La Unión Europea está también comprometida a fomentar la educación permanente de todos sus ciudadanos. Con respecto a las TIC se han llevado a cabo propuestas para su fomento en todos los estados como eEurope o i2010.

Actualmente la Comisión Europea ha propuesto una estrategia política llamada "Europa 2020" con el objetivo de cumplirlo en el citado año. Dentro de esta estrategia podemos encontrar la "agenda digital para Europa". Esta agenda digital propone explotar mejor el potencial de las TIC para favorecer la innovación, el crecimiento económico y el progreso. Uno de sus objetivos principales es la alfabetización en el

[6] Información obtenida de la página web (http://www.unesco.org/new/es/communication-and-information/resources/multimedia/photo-galleries/world-summit-on-the-information-society-wsis/, 02/05/2014)

entorno digital, y destaca la importancia de conocimientos de TIC para el futuro mercado laboral[7].

Por otra parte, la rama europea de Association for Computing Machinery (ACM Europe) realizó un informe analizando la situación de las TIC en la educación a nivel europeo de donde surgieron las siguientes recomendaciones a seguir por los estados miembros:

- Todos los ciudadanos europeos necesitan recibir educación en competencias digitales y en informática.
- La competencia digital (o alfabetización digital) abarca el uso fluido de herramientas informáticas y de Internet.
- La informática es la ciencia que está detrás de las TIC, es una ciencia distinguible de las demás y tiene sus propios conceptos y métodos. Es, al igual que las matemáticas, un área multidisciplinar que está detrás de los progresos actuales en ciencia, ingeniería y economía.
- La informática es un gran facilitador del progreso tecnológico actual y será clave para el futuro de la economía europea.
- Aunque la competencia digital está implantándose ampliamente en los currículos de los estados europeos, la mayoría carece de una adecuada base de educación informática.
- Europa debería de tomar medidas a este respecto para dejar a las generaciones futuras de ciudadanos en desventaja y para no crear una sociedad de meros consumidores de TIC.
- Los estudiantes deberían de beneficiarse de la informática como una asignatura independiente y desde edades tempranas.
- Se debería de ofertar un programa de formación de docentes en TIC a gran escala[8].

[7] Información obtenida de la página web (http://europa.eu/legislation_summaries/information_society/strategies/si0016_es.htm, 02/05/2014)
[8] Información obtenida de la página web (http://europe.acm.org/iereport/index.html, 02/05/2014)

A nivel estatal destacamos dos planes, el Plan Avanza2 y Escuela 2.0. El Plan Avanza2, desarrollado por Red.es (entidad pública que trabaja para que la sociedad española aproveche al máximo el potencial de Internet y las nuevas tecnologías), se concentra en proveer y mejorar las infraestructuras necesarias para conectarse a la red a toda la ciudadanía y en fomentar el uso de las TIC a nivel general. En el ámbito de la educación el plan tiene como objetivo potenciar la aplicación de las TIC al sistema educativo y formativo[9].

El programa Escuela 2.0, desarrollado por el Instituto Nacional de Tecnologías Educativas y Formación del Profesorado (INTEF) y suspendido por falta de presupuesto, proponía lo siguiente:

- Aulas digitales. Dotar de recursos TIC a los alumnos y alumnas y a los centros: ordenadores portátiles para alumnado y profesorado y aulas digitales con dotación eficaz estandarizada.
- Garantizar la conectividad a Internet y la interconectividad dentro del aula para todos los equipos. Posibilidad de acceso a Internet en los domicilios de los alumnos/as en horarios especiales.
- Promover la formación del profesorado tanto en los aspectos tecnológicos como en los aspectos metodológicos y sociales de la integración de estos recursos en su práctica docente cotidiana.
- Generar y facilitar el acceso a materiales digitales educativos ajustados a los diseños curriculares tanto para profesores y profesoras como para el alumnado y sus familias.
- Implicar a alumnos y alumnas y a las familias en la adquisición, custodia y uso de estos recursos[10].

El Ministerio de Educación, Cultura y Deporte también ha propuesto unas líneas de trabajo y de actuación en el ámbito de las TIC en educación para los próximos años

[9] Información obtenida de la página web (https://www.planavanza.es/InformacionGeneral/Estrategia2011/Paginas/Estrategia2011_2015.aspx, 02/05/2014)
[10] Información obtenida de la página web (http://www.ite.educacion.es/es/escuela-20, 02/05/2014)

dentro del Plan de Cultura Digital en la Escuela, aunque por el momento está en fase de proyecto para trabajar conjuntamente con las comunidades autónomas.

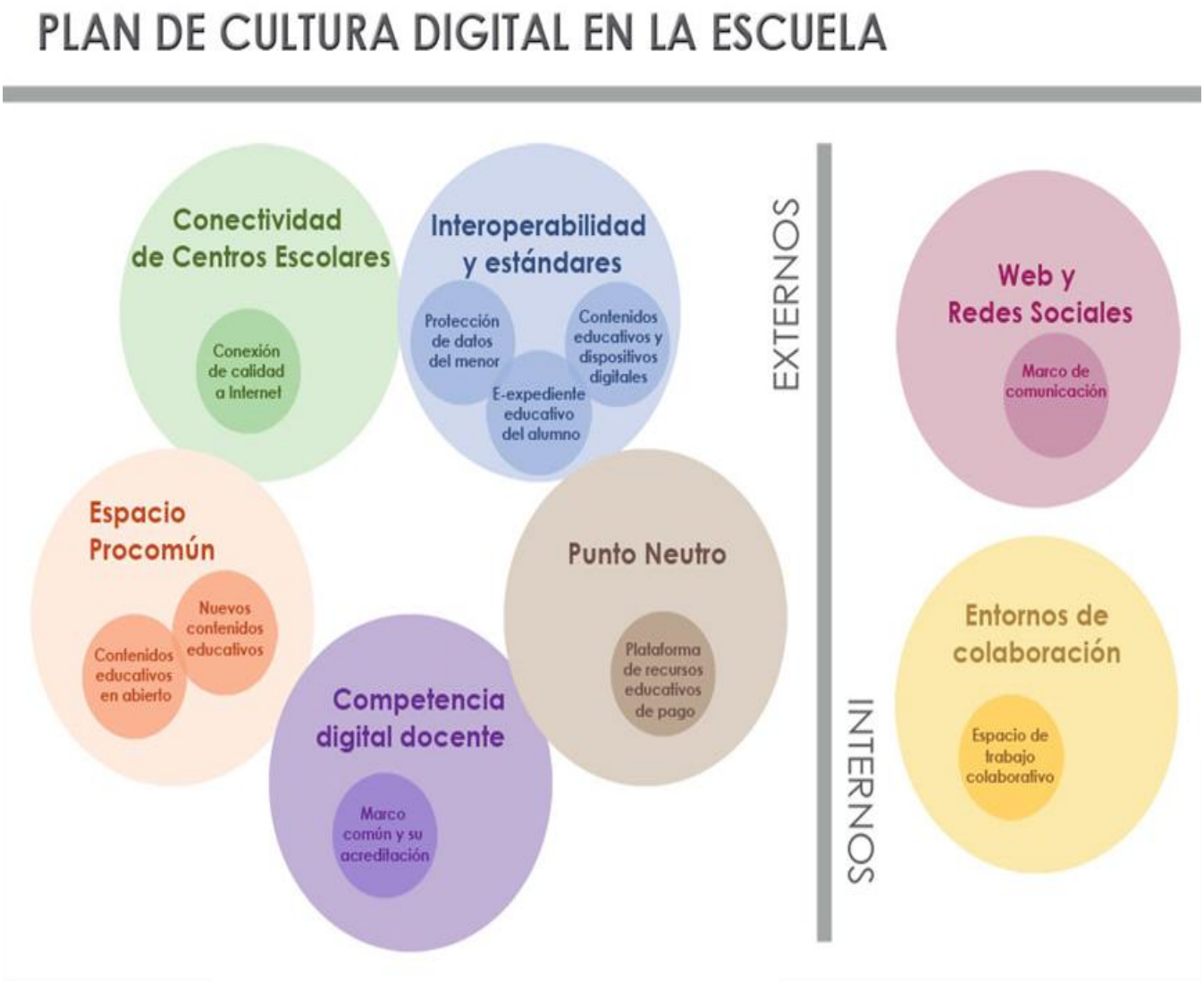


Figura 4. Plan de cultura digital en la escuela. Obtenida de la página web (http://blog.educalab.es/intef/2013/04/16/plan-de-cultura-digital-en-la-escuela/, 02/05/2014)

1.2.2 TIC en Educación Primaria en Navarra

En Navarra, al igual que ha ocurrido a nivel estatal, la falta de presupuesto ha influido en la cantidad y la calidad de proyectos para fomentar el uso de las TIC, aunque la intención del Departamento de Educación sigue siendo la de apostar por la formación en TIC, tanto para el alumnado como para el profesorado.

Prueba de ello es el Programa de Nuevas Tecnologías y Educación (PNTE), que engloba todos los objetivos, acciones, recursos y servicios que el Departamento de Educación del Gobierno de Navarra desarrolla en el ámbito de las TIC, dotando a los centros educativos navarros de todo lo necesario para funcionar conviviendo con las TIC.

Entre los contenidos más relevantes del PNTE, destacamos los siguientes:

- PNTE Apps: son los nuevos servicios del PNTE que a partir del curso 2012-2013 hacen uso de la tecnología de Google Apps para aplicarlo al ámbito de la educación. Engloban múltiples aplicaciones útiles para el funcionamiento y la organización en la escuela, como el calendario, e-mail, trabajo en la nube, etc.
- Formación en TIC: el PNTE ofrece su propio programa de formación con cursos online que se integran en dicho programa.
- Proyectos: incluye las convocatorias de proyectos de Nuevas Tecnologías que anualmente lleva a cabo el PNTE y los elementos clave del proyecto Integra TIC/IKT. Las convocatorias de proyectos de Nuevas Tecnologías pretenden que los centros educativos propongan ideas innovadoras para el fomento de las TIC. El proyecto Integra TIC/IKT, creado a partir del anteriormente citado proyecto Escuela 2.0, tiene el claro objetivo de la renovación tecnológica de las aulas. Se comenzó usando las tablets de manera experimental y se siguió instalando pizarras digitales en 3º ciclo de Educación Primaria.
- Servicios que se ofrecen a los docentes y que no están incluidos en PNTE Apps: webs dinámicas personales, blogs, etc.
- Servicios que se ofrecen a los centros educativos: soporte de redes informáticas, contrato de mantenimiento de equipos informáticos, webs dinámicas o software antivirus[11].

[11] Información obtenida de la página web (http://www.educacion.navarra.es/web/pnte/que-es-el-pnte, 03/05/2014)

Además del PNTE, otras instituciones como el Centro de Apoyo al Profesorado o la Universidad Pública de Navarra ofrecen apoyo y soporte a los centros educativos navarros en materia de TIC.

Por otra parte, el Real Decreto 132/2010, de 12 de febrero, por el que se establecen los requisitos mínimos de los centros que impartan las enseñanzas del la Educación Primaria, garantiza que en los centros todos los espacios en los que se desarrollen acciones docentes contarán con acceso a las TIC en cantidad y calidad adecuadas al número de puestos escolares[12], por lo que nuestra propuesta podría ser llevada a cabo en la totalidad de los centros escolares de Navarra.

[12] Información obtenida de la página web (http://www.cece.gva.es/re/docs/normativa/boe_val.pdf, 03/05/2014)

2 MARCO TEÓRICO: FUNDAMENTACIÓN E IMPLICACIONES

En este segundo punto analizaremos el proyecto Scratch desde el punto de vista educativo, dotando así a nuestra propuesta de un marco teórico en el que se incluyen las implicaciones prácticas y las referencias legales.

2.1. Programación informática y educación

De no ser por las adaptaciones anteriormente presentadas, la enseñanza de la programación informática se reduciría a la educación superior. Sin embargo, el esfuerzo de diferentes equipos de investigación ha propiciado la inserción de esta materia en niveles más elementales de la educación (por el momento hasta la Educación Primaria), creando nuevos contenidos y nuevos retos. Resulta necesario reflexionar sobre qué teorías educativas pueden sustentar la enseñanza de la programación informática en estos niveles, así como qué supone para el docente o cómo se puede integrar en las actuales leyes educativas.

2.1.1 Teorías del aprendizaje

Las dos teorías del aprendizaje donde la programación informática podría encajar son el constructivismo y el construccionismo, que como veremos más adelante están estrechamente relacionadas entre sí. Analizaremos las obras de Jean Piaget y de Seymour Papert, desarrolladores de las dos teorías respectivamente.

Según Hartle, Baviskar y Smith (2012) el constructivismo es una de las teorías del aprendizaje más aplicadas en la educación moderna. Definen el constructivismo como una teoría del aprendizaje en el que el alumno construye su conocimiento integrando, replanteando o cambiando sus estructuras cognitivas previas con las nuevas ideas o experiencias obtenidas. Esta construcción de conocimiento supone una implicación activa del alumnado en el proceso de aprendizaje, lo que requiere que las actividades sean motivadoras.

El uso de Scratch en el aula se adapta a esta definición del constructivismo ya que el alumnado participa activamente en el aprendizaje, cuestiona sus estructuras cognitivas previas y es altamente motivador.

Hartle et al. (2012) proponen también cuatro criterios para identificar una actividad educativa constructivista:

- Conocimiento previo: la actividad obliga al alumnado a sonsacar y a utilizar sus ideas preconcebidas.
- Disonancia cognitiva: la actividad plantea cuestiones que confrontan el conocimiento previo del alumnado.
- Aplicación del nuevo conocimiento: la actividad ayuda a integrar el nuevo conocimiento en la estructura cognitiva del alumnado.
- Reflexión sobre el aprendizaje: la actividad fomenta la reflexión del alumnado sobre su propio aprendizaje del nuevo conocimiento.

Si introducimos contenidos de programación informática en Educación Primaria veremos que primero debemos activar el conocimiento previo del alumnado sobre las TIC, y plantearles la cuestión de cómo se crean los programas informáticos y enseñarles a hacerlo. Esto creará disonancias cognitivas en el alumnado ya que o bien tendrán ideas preconcebidas de cómo funciona un programa o bien no se habrán planteado estas cuestiones antes. El entorno de Scratch, al ser adaptado, de manejo sencillo y estéticamente agradable, ayudará al alumnado a interiorizar los nuevos conocimientos. El objetivo de la programación informática es crear un programa, por lo que el resultado del proceso de aprendizaje será fácilmente apreciable para el alumnado, al ver el programa en funcionamiento podrán observar y reflexionar sobre los conocimientos adquiridos.

Siguiendo el modelo constructivista, Seymour Papert (miembro del equipo creador de Logo) propuso la teoría construccionista. El construccionismo acepta la idea constructivista de que el aprendizaje es un proceso activo donde el alumnado construye activamente su propio conocimiento del mundo a partir de las diferentes experiencias vividas.

Según Bruckman y Resnick (1995) el construccionismo propone que la construcción de conocimiento es particularmente efectiva cuando el alumnado está inmerso en la construcción de objetos o creaciones que son significativas. Pueden construir tanto objetos como castillos de arena o estructuras con bloques de madera como objetos

virtuales o programas informáticos. Lo importante es que estén inmersos en construir algo que sea significativo tanto para ellos mismos como para el entorno que les rodea.

Podemos ver cómo el proyecto Scratch encaja en la teoría construccionista, ya que al usar este programa el alumnado se sumerge en la construcción de un programa informático que será significativo para ellos y para su entorno, y en el proceso de creación estarán construyendo su propio conocimiento de manera eficaz.

En ambas teorías del aprendizaje subyace la idea de evitar ver la educación como un medio de transmisión de conocimiento en la que el alumnado es un receptor pasivo. La educación es entendida como un proceso activo por parte del alumnado en el que los educadores actúan de mediadores del aprendizaje, brindado a los alumnos situaciones y materiales significativos y motivadores.

2.1.2 Enfoques contemporáneos

Actualmente existen múltiples teorías con respecto a las TIC y la educación, aunque la mayoría de ellas tienen en común el fomento del uso de las nuevas tecnologías y la necesidad de cambiar de paradigma educativo, ya que nuestra sociedad se encuentra en constante avance hacia lugares hasta ahora desconocidos como la robótica, la nanotecnología o la mecánica cuántica que requieren de ciudadanos con conocimientos previos de informática para poder seguir avanzando.

Uno de los enfoques contemporáneos con respecto a la educación informática es el ver el pensamiento computacional como una competencia diferenciada. Según Wing (2008) el pensamiento computacional será una competencia fundamental en todo el mundo para mediados del siglo XXI, al igual que lo es el leer, escribir o la aritmética.

Wing (2008) define el pensamiento computacional como la habilidad de resolución de problemas, diseño de sistemas y comprensión del comportamiento humano basado en conceptos y procesos informáticos, y tiene semejanzas con el pensamiento analítico. Los dos procesos fundamentales son la abstracción y la automatización. El pensamiento computacional supone operar con múltiples niveles de abstracción

simultáneamente, definiendo la relación entre los diferentes niveles, y automatizar o mecanizar esos niveles de abstracción y sus relaciones.

Por otra parte, en los últimos años hemos podido ver como varias instituciones educativas ofrecen cursos en la red, que es lo que llamaríamos aprendizaje electrónico o e-learning. Sin embargo, hay un enfoque pedagógico más reciente llamado Computer Supported Collaborative Learning (CSCL) que va más allá del aprendizaje electrónico añadiendo la colaboración en línea (en forma de comunidad virtual) tanto del profesor y el estudiante como de los estudiantes entre sí y el crear y compartir el conocimiento.

Como Resta y Laferrière (2007) exponen, este aprendizaje colaborativo basado en el ordenador facilita la comunicación tanto sincrónica como asincrónica, en grupos de trabajo reducidos o numerosos y desde lugares geográficamente distanciados. No se trata de que este tipo de aprendizaje sustituya a la educación presencial, sino de utilizarlas conjuntamente aprovechando las ventajas de cada uno. Además, distinguen las siguientes características:

- El alumnado se responsabiliza de su aprendizaje.
- La enseñanza y el aprendizaje son experiencias compartidas.
- El alumnado reflexiona sobre su proceso de aprendizaje.
- Se desarrollan habilidades sociales y de grupo al tener que dar y recibir y en la búsqueda de consenso.
- El aprendizaje es un proceso activo donde el docente tiene el rol de facilitador.

El proyecto Scratch encaja con estos enfoques contemporáneos ya que al utilizarlo el estudiante está desarrollando el pensamiento computacional y su plataforma permite la colaboración en línea tanto con los compañeros como con estudiantes de cualquier parte del mundo. También encaja en la mayoría de las teorías contemporáneas con respecto a la educación dadas sus características de colaboración, creación, desarrollo de la competencia digital o su carácter multidisciplinar.

2.1.3 Aportaciones de la programación informática a la educación

Según Pea y Kurland (1984) la programación informática tiene múltiples beneficios para la educación en todos los niveles, concretamente estudiaron las siguientes seis aportaciones al desarrollo cognitivo:

- Habilidades matemáticas: aunque la programación informática desarrolla la capacidad cognitiva general, ayuda especialmente a las habilidades matemáticas, ya que la informática ha estado desde siempre unida a esta disciplina (los primeros ordenadores se crearon para realizar cálculos matemáticos). Sin embargo, la destreza matemática no es un requisito imprescindible para aprender a programar.
- Capacidad de memoria y de concentración: la programación informática es una actividad que requiere un uso intensivo de la memoria y una gran concentración, ya que se manejan diferentes variables a la vez y hay que prestar atención a los detalles (una letra mal puesta puede significar que el programa no funcione correctamente). Así, el alumno estará desarrollando su capacidad de memoria y de concentración.
- Habilidades de razonamiento analógico: el alumnado que aprenda a programar podrá ver analogías entre la forma de programar y la resolución de problemas en otros aspectos de su vida. También podrá utilizar sus conocimientos previos a la hora de programar. La programación informática ayuda así a desarrollar el razonamiento analógico, fomentando el uso interdisciplinar de sus habilidades.
- Habilidades de razonamiento condicional: las frases condicionales son una parte fundamental de la programación (e.g., si se presiona la tecla “W”, entonces ocurrirá que el objeto “x” se desplace hacia arriba), por lo que la programación informática ayudará al alumnado a desarrollar su razonamiento condicional a través de conexiones lógicas de conjunción, negación o disyunción.
- Habilidades de pensamiento procedimental: al programar el alumnado tiene que redactar instrucciones precisas y complejas que luego el programa ejecutará, por lo que están desarrollando su pensamiento procedimental que les ayudará en diferentes aspectos de la vida que requieran dar o ejecutar

instrucciones en un orden determinado, desde cocinar siguiendo una receta hasta ayudar a un turista a decirle por dónde se va al museo.

- Habilidades de razonamiento temporal: estas habilidades van más allá de la comprensión temporal de la secuencia de acontecimientos. Al estar programando hay instrucciones dentro de otras instrucciones y comandos que se solapan unos a otros, y esto requiere un control preciso de los tiempos del programa. Son habilidades complicadas de adquirir antes de los 7 u 8 años pero que influyen notablemente en el desarrollo del razonamiento temporal.

Por otra parte, Badger (2009) afirma que Scratch aporta los siguientes conceptos de programación informática al alumnado:

- Secuencia: para crear un programa en Scratch, se requiere pensar sistemáticamente sobre el orden de los pasos.
- Iteración (ciclos): los comandos “por siempre” y “repetir” se utilizan para crear iteraciones (repetición de una serie de instrucciones).
- Variables: las variables sirven para almacenar números o cadenas de caracteres (palabras). Las instrucciones correspondientes a variables permiten crearlas y usarlas en un programa. Scratch admite tanto variables globales (para todos los objetos) como especificas para un solo objeto.
- Sentencias condicionales: los comandos “si” y “si no” verifican si una proposición simple o compuesta es verdadera (si se cumple una condición).
- Entradas vía teclado: los comandos “preguntar” y “esperar” solicita a los usuarios escribir algo mediante el teclado. El comando “respuesta” es una variable que almacena lo último que se ingresó vía teclado.
- Manejo de eventos: los comandos “al presionar tecla” y “al presionar objeto” son ejemplos del manejo de eventos. Estas instrucciones de control responden a eventos provocados por el usuario o por otra parte del programa.
- Hilos (paralelismo): poner en marcha dos pilas de instrucciones al mismo tiempo hace que se creen dos hilos independientes que se ejecutan en paralelo.
- Números al azar: el comando “número al azar entre” selecciona un número entero dentro de un rango dado.

- Lógica booleana: Scratch permite trabajar con el álgebra booleana (que utiliza las operaciones “y”, “o” y “no”). Las proposiciones compuestas se forman con dos o más proposiciones sencillas unidas por operadores lógicos.
- Diseño de interfaz de usuario: con Scratch se diseñan interfaces de usuario interactivas. Por ejemplo, usar objetos para que funcionen como botones. Por ejemplo, al hacer clic sobre un objeto llamado “lápiz” se ejecutará un conjunto de instrucciones.
- Coordinación y sincronización: el comando “enviar a todos” manda un mensaje a todos los objetos y espera a que se ejecuten las acciones de los objetos activados. El comando “al recibir” coordina acciones de diferentes objetos. Este par de instrucciones permiten la sincronización.
- Listas (arreglos): las listas son un tipo de estructura de datos que puede considerarse un arreglo bidimensional de “n x 1”. Con varias listas se puede conformar una matriz (arreglo bidimensional de n x m).
  Las instrucciones correspondientes a “listas” permiten almacenar y acceder a arreglos de números o cadenas de caracteres.
- Interacción dinámica: los comandos “x del ratón”, “y del ratón” y “volumen del sonido” se pueden utilizar como entrada dinámica para interactuar en tiempo real con los programas de Scratch.

El proceso de llevar a la acción estos conceptos de programación se apoya en una serie particular de prácticas computacionales. Estas prácticas de solución de problemas incluyen:

- Incremental e iterativo: desarrollar un segmento o parte de un programa, ponerlo a prueba (ensayarlo) y luego desarrollar otro segmento de ese mismo.
- Probar y depurar: Asegurarse de que las cosas funcionen y localizar y arreglar errores.
- Reusar o remezclar: Hacer algo en base a lo que otros o el propio alumno han hecho.

- Abstracción y modularización: Construir algo grande juntando colecciones de partes pequeñas[13].

Por último, López García (2009) propone varios beneficios de la programación informática para la educación escolar, concretamente para la comprensión de los siguientes conceptos matemáticos:

- Concepto de variable: una variable es una ubicación de memoria en el ordenador que tiene un nombre (identificador) y en la que se pueden almacenar diferentes valores.
- Concepto de función: el alumnado puede crear procedimientos que se comportan como funciones (aceptan parámetros, realizan cálculos y reportan un resultado).
- Manejo de ecuaciones y gráficos.
- Modelado matemático: algunas de las ideas clave de los modelos matemáticos están presentes en los manipulables virtuales (simulaciones y micro mundos). Estos manipulables se pueden emplear tanto en procesos de entrenamiento como de educación matemática. Sin embargo, la tendencia es a utilizarlos en ambientes en los que el alumnado se convierte en diseñador y no en simple consumidor.

2.1.4 Scratch en Educación Primaria

Como hemos podido observar el uso de Scratch en educación está avalado por diferentes teorías del aprendizaje y enfoques actuales, pero la cuestión que planteamos es su adecuación en un aula de Educación Primaria.

Wilson y Moffat (2010) realizaron un estudio para evaluar el uso de Scratch en un aula real de Educación Primaria, ya que veían que los estudios realizados hasta entonces evaluaban el uso del programa en actividades extraescolares, diferenciado del sistema educativo. Para el estudio escogieron una clase de 21 alumnos (completamente principiantes en programación informática) de entre 8 y 9 años de edad de una escuela

[13] Información obtenida de la página web (http://www.eduteka.org/modulos/9/285/1206/1, 04/05/2014)

de un barrio económicamente desfavorecido de Glasgow (Escocia). Utilizaron la sesión de informática (una semanal) durante 8 semanas para introducir el programa Scratch desde cero al alumnado.

Los resultados obtenidos tanto por parte del alumnado como del profesorado fueron satisfactorios. El alumnado mostró cierto avance cognitivo con respecto a la clase de informática usual de la escuela, y Wilson y Moffat (2010) sugieren que en 8 sesiones es difícil ver una mejoría significativa, pero que con más tiempo se podría observar un avance mayor. El cambio que sí se produjo y se pudo observar fue en el plano afectivo, ya que el alumnado se mostró mayoritariamente motivado y cómodo en el aula.

Algunas de las conclusiones de este estudio fueron que Scratch es capaz de involucrar a todo tipo de alumnado, que puede ayudar al desarrollo cognitivo de los niños, que desarrollan sus habilidades sociales al compartir y valorar sus creaciones o que el alumnado de Primaria está preparado para adquirir conceptos de programación informática. Cabe mencionar la entrevista al tutor en la que expresa su sorpresa al ver cuán rápidamente se adaptaron sus alumnos al entorno de Scratch y lo mucho que les gustaban las clases.

Sin embargo algunas de las funcionalidades de Scratch resultaban excesivamente complejas para el alumnado, por lo que nuestra conclusión a este respecto es que introducir Scratch en Educación Primaria es factible, pero graduando por niveles de dificultad según las habilidades cognitivas de la edad de los estudiantes.

Esto se puede llevar a cabo introduciendo las funcionalidades de Scratch gradualmente, desde las funciones más básicas en primer curso hasta las más complejas en sexto curso de Primaria, como veremos más adelante. Esto supondrá que las creaciones de los alumnos también serán más complejas a medida que avancen de curso.

2.1.5 Implicaciones prácticas

El uso de Scratch en un aula de Educación Primaria implica múltiples cuestiones a tener en consideración, que mostramos en la siguiente tabla de elaboración propia:

Tabla 2. Implicaciones prácticas del uso de Scratch en el aula de Primaria

Aspecto	**Implicaciones prácticas**
Espacio	Dependiendo de los recursos informáticos de la escuela se podrá utilizar Scratch en el aula o en el aula de informática. Como la programación informática requiere dedicar tiempo a estar sentado y concentrado mirando la pantalla del equipo informático, se debe adecuar la iluminación del aula, la ventilación (ya que los equipos informáticos desprenden calor) y fomentar una correcta postura al sentarse.
Material	Pueden ser ordenadores de sobremesa o portátiles, y el equipo de Scratch está trabajando para poder utilizarlo también en tablets. Para sacar el mayor provecho de Scratch es recomendable disponer de una buena conexión a la red. Hay que tener en cuenta que si no se disponen de equipos informáticos suficientes para todo el alumnado, teniendo que trabajar por parejas o en grupos pequeños, el modo de introducir Scratch varía notablemente, ya que debe de buscar consensos antes de ponerse a programar.
Tiempo	Si se utiliza Scratch como un recurso informático separado de las otras asignaturas, se podría introducir en la sesión semanal correspondiente en el aula de informática, empezando desde primer curso de Educación Primaria. Sin embargo las posibilidades de Scratch permiten el trabajo conjunto con todas las asignaturas, por lo que podría utilizarse más a menudo como material complementario en cualquier asignatura.
Grupos	Al estar involucrados en la comunidad en línea de Scratch, no es necesario que el alumnado trabaje en grupo previamente, por lo que lo más recomendable sería que cada alumno disponga de un equipo informático para su uso individual.
Metodología	Se pueden distinguir tres fases en el trabajo con Scratch. En la primera fase el alumnado aprende y se familiariza con el manejo de las funcionalidades de Scratch. En la segunda fase el alumnado crea sus propios programas haciendo que desarrolle sus habilidades de programación. En la tercera fase publica su trabajo y lo comparte con la comunidad virtual, sintiendo valorado su trabajo y valorando también el trabajo de los demás, tanto a través de los usuarios de la comunidad online como de sus compañeros de aula.
Rol del docente	Al introducir Scratch al alumnado, el primer papel que toma el docente es el de instructor de las funcionalidades del programa.

	Una vez que el alumnado se maneja fluidamente el rol del profesor es el de facilitador, ayudante, motivador y guía del proceso de programación.
Motivación	Hay que tener en cuenta que la plataforma Scratch por sí misma puede motivar al alumnado, por lo que el docente tendrá que ayudar a que esa motivación no decaiga, proponiendo nuevas posibilidades de trabajo o colaboración con la comunidad online.
Formación previa	El maestro que vaya a introducir Scratch en su aula deberá contar con una formación previa en el manejo del programa para ofrecer una docencia adecuada y eficaz. Además el Ministerio de Educación, Cultura y Deporte está elaborando el Marco Común de Competencia digital docente en el que se definen las habilidades a adquirir por parte del docente[14].

Estas implicaciones se tendrán que tener en cuenta antes de la implantación de Scratch en la escuela, y se deberá de contar con el apoyo del centro para introducirlo de manera sistematizada, organizada y gradual. Asimismo será necesaria una coordinación entre el profesorado.

2.2. Relación con el currículo oficial de Navarra

La educación navarra está actualmente regulada por la Ley Orgánica para la Mejora de la Calidad Educativa (LOMCE), y el currículo de Educación Primaria está definido por el Decreto Foral 24/2007. En ambas podemos encontrar múltiples referencias a las TIC y a la competencia digital, por lo que a continuación mencionaremos las más relevantes para nuestra propuesta.

Según podemos leer en la Ley Orgánica para la Mejora de la Calidad Educativa:

"La incorporación generalizada al sistema educativo de las Tecnologías de la Información y la Comunicación (TIC), que tendrán en cuenta los principios de diseño para todas las personas y accesibilidad universal, permitirá personalizar la educación y adaptarla a las necesidades y al ritmo de cada alumno o alumna. Por una parte, servirá para el refuerzo y apoyo en los casos de bajo rendimiento y, por otra, permitirá expandir sin limitaciones los conocimientos transmitidos en el aula. Los alumnos y alumnas con motivación podrán así acceder, de acuerdo con su capacidad, a los

[14] Información obtenida de la página web (http://blog.educalab.es/intef/2014/02/21/jornada-de-trabajo-sobre-marco-comun-de-competencia-digital-docente/, 05/05/2014)

recursos educativos que ofrecen ya muchas instituciones en los planos nacional e internacional. Las TIC serán una pieza fundamental para producir el cambio metodológico que lleve a conseguir el objetivo de mejora de la calidad educativa. Asimismo, el uso responsable y ordenado de estas nuevas tecnologías por parte de los alumnos y alumnas debe estar presente en todo el sistema educativo. Las TIC serán también una herramienta clave en la formación del profesorado y en el aprendizaje de los ciudadanos a lo largo de la vida, al permitirles compatibilizar la formación con las obligaciones personales o laborales y, asimismo, lo serán en la gestión de los procesos[15].”

Quedan claramente reflejadas en esta ley las necesidades de cambio y de utilización y fomento de las TIC, por lo que el uso de Scratch en el aula queda avalado legalmente.

Por otra parte, en el currículo de Educación Primaria de Navarra, donde encontramos la competencia digital como una competencia básica, podemos leer:

“Esta competencia consiste en disponer de habilidades para buscar, obtener, procesar y comunicar información, y para transformarla en conocimiento. Incorpora diferentes habilidades, que van desde el acceso a la información hasta su transmisión en distintos soportes una vez tratada, incluyendo la utilización de las tecnologías de la información y la comunicación como elemento esencial para informarse, aprender y comunicarse.

Está asociada con la búsqueda, selección, registro y tratamiento o análisis de la información, utilizando técnicas y estrategias diversas para acceder a ella según la fuente a la que se acuda y el soporte que se utilice (oral, impreso, audiovisual, digital o multimedia). Requiere el dominio de lenguajes específicos básicos (textual, numérico, icónico, visual, gráfico y sonoro) y de sus pautas de decodificación y transferencia, así como aplicar en distintas situaciones y contextos el conocimiento de los diferentes tipos de información, sus fuentes, sus posibilidades y su localización, así como los lenguajes y soportes más frecuentes en los que ésta suele expresarse.

[15] Ley Orgánica 8/2013, de 9 de diciembre, para la mejora de la calidad educativa.

Disponer de información no produce de forma automática conocimiento. Transformar la información en conocimiento exige de destrezas de razonamiento para organizarla, relacionarla, analizarla, sintetizarla y hacer inferencias y deducciones de distinto nivel de complejidad; en definitiva, comprenderla e integrarla en los esquemas previos de conocimiento. Significa, asimismo, comunicar la información y los conocimientos adquiridos empleando recursos expresivos que incorporen, no sólo diferentes lenguajes y técnicas específicas, sino también las posibilidades que ofrecen las tecnologías de la información y la comunicación.

Ser competente en la utilización de las tecnologías de la información y la comunicación como instrumento de trabajo intelectual incluye utilizarlas en su doble función de transmisoras y generadoras de información y conocimiento. Se utilizarán en su función generadora al emplearlas, por ejemplo, como herramienta en el uso de modelos de procesos matemáticos, físicos, sociales, económicos o artísticos. Asimismo, esta competencia permite procesar y gestionar adecuadamente información abundante y compleja, resolver problemas reales, tomar decisiones, trabajar en entornos colaborativos ampliando los entornos de comunicación para participar en comunidades de aprendizaje formales e informales, y generar producciones responsables y creativas.

La competencia digital incluye utilizar las tecnologías de la información y la comunicación extrayendo su máximo rendimiento a partir de la comprensión de la naturaleza y modo de operar de los sistemas tecnológicos, y del efecto que esos cambios tienen en el mundo personal y sociolaboral. Asimismo supone manejar estrategias para identificar y resolver los problemas habituales de software y hardware que vayan surgiendo. Igualmente permite aprovechar la información que proporcionan y analizarla de forma crítica mediante el trabajo personal autónomo y el trabajo colaborativo, tanto en su vertiente sincrónica como diacrónica, conociendo y relacionándose con entornos físicos y sociales cada vez más amplios. Además de utilizarlas como herramienta para organizar la información, procesarla y orientarla para conseguir objetivos y fines de aprendizaje, trabajo y ocio previamente establecidos.

En definitiva, la competencia digital comporta hacer uso habitual de los recursos tecnológicos disponibles para resolver problemas reales de modo eficiente. Al mismo tiempo, posibilita evaluar y seleccionar nuevas fuentes de información e innovaciones tecnológicas a medida que van apareciendo, en función de su utilidad para acometer tareas u objetivos específicos.

En síntesis, el tratamiento de la información y la competencia digital implican ser una persona autónoma, eficaz, responsable, crítica y reflexiva al seleccionar, tratar y utilizar la información y sus fuentes, así como las distintas herramientas tecnológicas; también tener una actitud crítica y reflexiva en la valoración de la información disponible, contrastándola cuando es necesario, y respetar las normas de conducta acordadas socialmente para regular el uso de la información y sus fuentes en los distintos soportes[16].”

Podemos ver cómo la competencia digital no se limita a ser usuario de las TIC sino que implica el desarrollo integral de la persona y que haga propias las TIC para lograr sus objetivos.

[16] Decreto Foral 24/2007, de 19 de marzo, por el que se establece el currículo de las enseñanzas de Educación Primaria en la Comunidad Foral de Navarra.

3 MATERIAL Y MÉTODOS

3.1 El proyecto Scratch

3.1.1 Definición y características

Scratch se puede definir como un lenguaje de programación y una comunidad en línea donde se pueden crear historias interactivas, juegos, actividades, animaciones, etc., y compartir las creaciones con cualquier persona en el mundo que disponga de conexión a internet. Al diseñar y programar proyectos de Scratch, los jóvenes y niños aprenden a pensar creativamente, razonar sistemáticamente y trabajar colaborativamente. Está disponible en varios idiomas, dispone de página de ayuda y de tutorial (integrado en el editor) y la versión 2 requiere de un navegador web reciente. La versión 1.4 es descargable para trabajar sin conexión y ambas versiones son gratuitas[17].

Scratch es el entorno de programación de computadores que más resultados ha cosechado en la educación escolar. Así lo demuestra la contundente cifra de 3.289.354 de proyectos elaborados con esta herramienta y publicados en su sitio web (http://scratch.mit.edu/), por 1.525.118 docentes y estudiantes de todo el mundo (datos del 28 de abril de 2013)[18].

Scratch se lanzó oficialmente en mayo del 2007 e inicialmente tuvo amplia acogida entre quienes venían trabajando con alguna de las versiones de Logo. Pero, en muy corto tiempo, su audiencia se amplió y consiguió cautivar a docentes de todo el planeta que comenzaron a usarlo en sus clases. A partir de mayo del 2013, habrá un cambio radical en el sitio web de Scratch, que desde esa fecha, desplegará la versión 2.0 de Scratch que funciona completamente en línea. Hasta ahora, en ese sitio, los usuarios podían subir los proyectos que habían elaborado con la versión 1.4 de la herramienta (solo descargable), crear galerías con esos proyectos, consultar los de otros usuarios, ver proyectos destacados por la comunidad de usuarios, etc. A partir del lanzamiento de la versión 2.0, se puede crear, editar y ver los proyectos

[17] Información obtenida de la página web (http://scratch.mit.edu/help/faq/, 7/05/2014)
[18] Información obtenida de la página web (http://www.eduteka.org/modulos/9/332/2158/1, 07/05/2014)

directamente en un navegador web, sin tener que descargar e instalar ningún programa en el ordenador.

Existen también extensiones del proyecto Scratch que añaden nuevas posibilidades, como el portal ScratchED, página web destinada a docentes para que compartan experiencias con Scratch, o Build Your Own Blocks (BYOB), programa que permite al usuario crear nuevos bloques personalizados para utilizar en Scratch.

Las posibilidades que ofrece Scratch son ilimitadas, dependiendo del grado de creatividad del alumnado, aunque Badger (2009) propone los siguientes usos didácticos del programa:

- Tarjeta interactiva: se pueden crear invitaciones para fiestas de cumpleaños, tarjetas de felicitación del día de la madre, etc., que se pueden enviar por correo electrónico u otros medios.
- Presentación de diapositivas: podemos hacer desde un álbum de fotos con funcionalidades visualmente atractivas hasta presentaciones de trabajos.
- Narración: una funcionalidad muy interesante de Scratch es el poder crear historias o cuentos animados e interactivos. Puede sernos realmente útil como docentes no sólo a la hora de crear las narraciones sino para utilizar esas producciones como material didáctico.
- Juegos: la creación de un juego funcional, con todo lo que ello conlleva (manejo, puntuaciones, jugabilidad, gráficos, variables, etc.), puede que sea una de las actividades más complejas que se pueden llevar a cabo con Scratch. Sin embargo el alumnado que consiga crear un juego que encandile a los demás se sentirá inmensamente satisfecho de su trabajo. Además la creación de un juego conlleva la constante revisión y depuración de errores y la aplicación de mejoras.
- Adivinación: para practicar las variables con el alumnado son adecuadas las creaciones de preguntas y respuestas que “adivinan” el futuro, además del interés que suscita a los demás.
- Problemas matemáticos: además de ejercicios de aritmética, Scratch permite crear problemas más complejos que incluyan gráficos, estadísticas o funciones.

También recuerda que las creaciones con Scratch están orientadas a la participación en la comunidad online, por lo que tienen licencia Creative Commons para compartirlas públicamente, permitiendo que otros usuarios modifiquen las creaciones a su gusto (lo que en la página web llaman "remixing").

En el Anexo I podemos encontrar una propuesta de 10 tipos de actividades para realizar con Scratch que pueden servir al docente como punto de partida o para proponer retos al alumnado. También nos sirve para ejemplificar el tema que estamos tratando en el presente trabajo.

3.1.2 Interfaz de usuario

Siguiendo la "Guía de referencia de Scratch 2.0"[19] analizaremos los cuatro elementos principales en la interfaz de Scratch, que son el menú, el fondo o escenario, los personajes u objetos y los comandos o instrucciones, además de las características adicionales que nos ofrece el programa.

El menú nos permite realizar las configuraciones generales del programa, así como abrir o guardar proyectos.

Figura 5. Menú de la interfaz de Scratch. Obtenida de la página web (http://scratch.mit.edu/projects/editor/?tip_bar=getStarted, 08/05/2014)

Como podemos ver en la figura 4 el primer icono, el logo de Scratch, nos envía a la página web del proyecto, donde podemos explorar las creaciones de los usuarios de la comunidad virtual.

El icono de la bola del mundo representa las opciones de lenguaje, para cambiar el idioma de la interfaz de usuario de Scratch. Cualquiera puede agregar o editar las traducciones de idioma de Scratch. En los idiomas actuales están incluidos tanto el

[19] Información obtenida de la página web (http://www.eduteka.org/pdfdir/ScratchGuiaReferencia.pdf, 08/05/2014)

inglés como el castellano, el euskera o el francés, lo que nos permite trabajar en la totalidad de los centros escolares de Navarra.

Al seleccionar “archivo” se puede crear un nuevo proyecto, guardar el proyecto actual, ir a “mis proyectos”, cargar un proyecto desde un archivo grabado en el ordenador, descargar un proyecto al ordenador y revertir los cambios realizados.

El menú “editar” ofrece varias opciones para editar el proyecto actual. “Deshacer borrado” permite recuperar el último bloque, programa, objeto, disfraz o sonido que se borró. “escenario pequeño” permite agrandar o encoger el área del escenario. El “modo turbo” permite ver la ejecución del programa paso a paso.

Desde el menú “Sugerencias” se puede acceder a la página de ayuda con enlaces a materiales de referencia, tutoriales y preguntas frecuentes. También se puede acceder a la página que contiene todas las pantallas de ayuda. “About” tiene una funcionalidad parecida, sólo que nos redirige a la página web de información y ayuda de Scratch.

Por último, los últimos cinco iconos correspondan a la “barra de herramientas”, y se aplican directamente en el proyecto. “Duplicar” crea una copia del objeto o comando que seleccionemos, “borrar” elimina el objeto o comando elegido, “crecer” y “encoger” cambian el tamaño de un objeto y “ayuda de bloques” nos redirige al apartado de ayuda del bloque que seleccionemos.

El fondo o escenario del proyecto que realicemos no es un elemento estático y simplemente decorativo, sino que tiene funcionalidades específicas que permiten interactuar con él.

El escenario tiene 480 puntos (pixeles) de ancho y 360 puntos de alto y está dividido en un plano cartesiano x-y. El centro del escenario corresponde a las coordenadas X:0, Y:0. Para encontrar posiciones x-y en el escenario, podemos mover el ratón en él y mirar la información de posición x-y del ratón, justo debajo del escenario. Esta información nos es fundamental a la hora de dar instrucciones de posiciones concretas a los objetos.

Al seleccionar el fondo podemos agregar comandos específicos para el escenario (más limitados que los de los objetos), y en la pestaña “fondo” podemos editar el escenario a nuestro gusto y añadir más fondos para el proyecto.

Los personajes u objetos, al igual que los escenarios, se pueden bien seleccionar de la biblioteca del programa o bien crear (desde cero o importando una imagen de una cámara o de cualquier otra fuente).

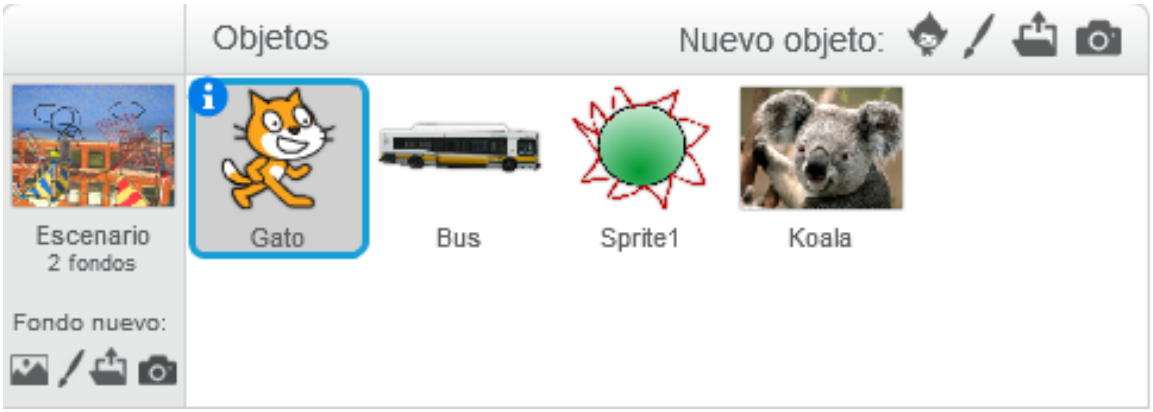


Figura 6. “Escenario” y “objetos” de la interfaz de Scratch. Obtenida de la página web (http://scratch.mit.edu/projects/editor/?tip_bar=getStarted, 08/05/2014)

En la figura 5 podemos observar los diferentes tipos de objetos. El gato es el objeto por defecto del programa (y la mascota), el autobús es un objeto realista de la biblioteca de imágenes del programa (a la biblioteca se entra a través del primer icono al lado de “nuevo objeto”), el sol verde es de creación propia con el editor de Scratch (al editor se accede a través del icono del pincel) y la imagen del koala es importada (se pueden importar desde el ordenador, que es el icono de la carpeta, o desde la cámara, que es el icono de la cámara).

Podemos observar también que el nombre de los objetos por defecto es “sprite 1” o el número que le corresponda, por ello resulta importante para trabajar con comodidad nombrar los objetos de manera identificativa. Al lado izquierdo tenemos las opciones de escenario, que son prácticamente idénticas a las de los objetos. Al seleccionar un objeto, al igual que los escenarios, se mostrarán los bloques de instrucciones o comandos asignados a ese objeto, por lo que a la hora de programar hay que tener en cuenta qué es lo que tenemos seleccionado.

Los bloques de comandos o instrucciones se arrastran con el ratón al espacio vacío, que es el editor de programación, y se encadenan para realizar acciones en una secuencia. También podemos seleccionar un grupo de bloques y duplicarlo, para no tener que programar las mismas instrucciones una y otra vez en cada objeto.

En la "paleta de bloques", donde encontramos todos los bloques que podemos utilizar, existen tres tipos principales de bloques:

- Bloques para "apilar": estos bloques tienen salientes y muescas en la parte inferior y superior respectivamente y pueden encajarse unos con otros para formar grupos. Algunos de estos bloques tienen un área de ingreso de información en su interior, en la que se puede escribir un número o seleccionar un elemento de un menú desplegable. Algunos bloques permiten la opción de apilar dentro del propio bloque.
- "Sombreros": estos bloques tienen redondeada la parte superior y se ubican en la parte superior de los grupos de bloques. Esperan a que suceda un evento, como por ejemplo que se presione una tecla y entonces ejecutan los bloques que están debajo de ellos.
- "Reporteros": este tipo de bloques están diseñados para encajar en el área de ingreso de información de otros bloques. Los reporteros con bordes redondeados reportan números o cadenas de texto y encajan en bloques que tienen espacios redondeados o rectangulares. Los reporteros con bordes en punta reportan valores booleanos y encajan dentro de bloques con espacios que terminan en punta o son rectangulares. Algunos de los bloques reporteros tienen una casilla enseguida de ellos. Si se hace clic en la casilla, aparece un monitor en el escenario, que muestra el valor actual del reportero. A medida que el valor del reportero cambia, el monitor se actualiza automáticamente.

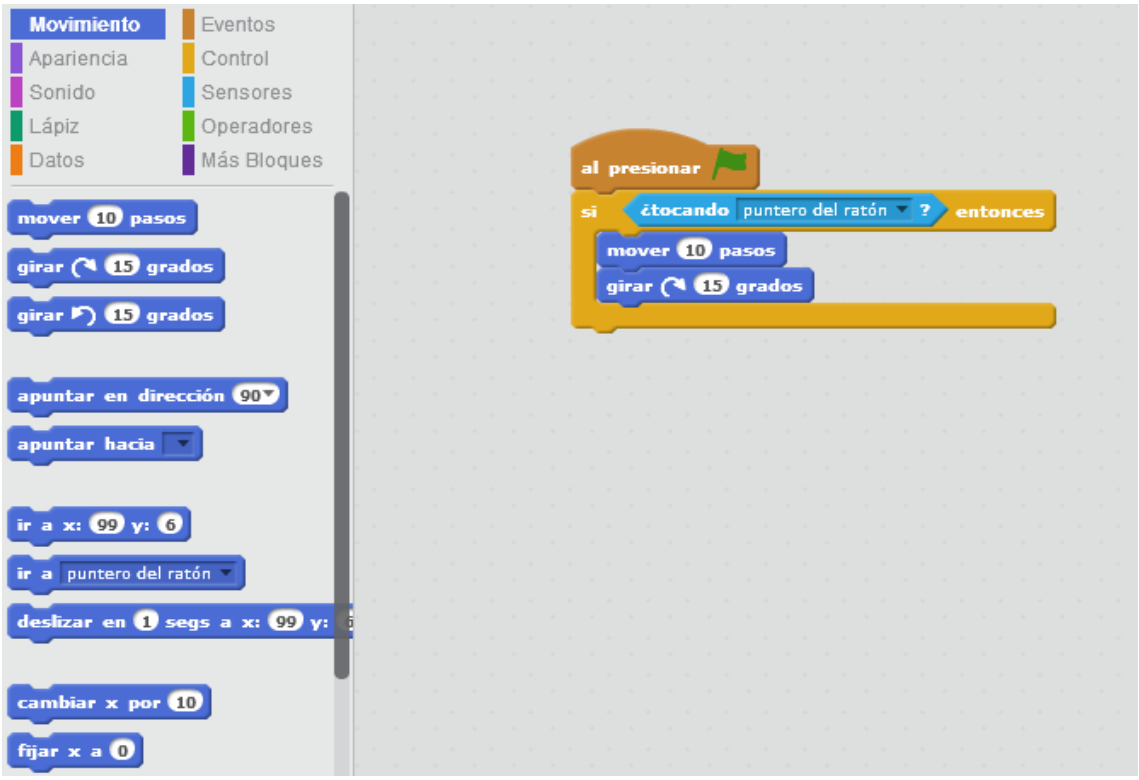


Figura 7. "Paleta de bloques" y los tres tipos de bloques en el editor (el naranja oscuro es "sombrero", el azul claro es "reportero" y el resto son de "apilar"). Obtenida de la página web (http://scratch.mit.edu/projects/editor/?tip_bar=getStarted, 08/05/2014)

Como podemos observar en la figura 7 los bloques están divididos en diez categorías diferentes, y la descripción de cada bloque se puede encontrar en el Anexo II.

Entre las características adicionales del programa encontramos:

- Banderas: sirven para iniciar o detener todos los comandos.
- Disfraces: permiten tener varias imágenes para un solo objeto.
- Sonidos: se pueden grabar o importar los sonidos que deseemos.
- Estilos de rotación: define cómo rotará cada objeto.
- Editor de pinturas: un editor básico de imágenes para editar objetos.
- Cámara: nos permite sacar fotos o interactuar con el programa.
- Mochila: sirve para guardar objetos, escenarios, disfraces o grupos de comandos para usar en diferentes proyectos
- Funciones de la comunidad online: seguir, valorar, etc. a otros usuarios.

3.2 Propuesta de estudio de caso

Nuestra propuesta de introducir contenidos de programación informática a través de Scratch en las escuelas de Educación Primaria de Navarra necesitaría demostrar su viabilidad a través de diferentes estudios.

En este trabajo proponemos la realización de un estudio de caso general aplicable a todos los cursos y edades de Primaria.

Comenzaremos por un objetivo general que será el de comprobar si el fomento del uso de las TIC como herramientas de producción y la adquisición de contenidos de programación informática a través de un lenguaje simplificado y visual (Scratch) son aplicables en Educación Primaria.

En las secciones anteriores hemos visto los antecedentes y hemos discutido su adecuación pedagógica, por lo que nos queda verificar en un aula si nuestro objetivo es cumplible. Para eso proponemos una metodología de trabajo y diferentes tipos de actividades que detallaremos a continuación.

Desde el punto de vista teórico proponemos una metodología constructivista, ya que como hemos visto es la que mejor se adecua al uso de Scratch. Esto significa que permitiremos al alumnado construir su conocimiento sobre el programa a través de su experiencia utilizándolo. El programa permite además que el alumnado aprenda de sus propios errores, ya que si existe algún error el resultado no será el esperado y el alumno tendrá que buscar dónde está el fallo y modificarlo. En este sentido el papel del profesor será necesariamente instructivo en la primera o primeras sesiones, pero a medida que el alumnado maneje el programa cambiará a guía, apoyo y motivador.

Con respeto al espacio, el tiempo y los materiales, nos queda claro que hace falta que cada alumno disponga de su propio ordenador, por lo que proponemos que las sesiones, una semanal durante un mínimo de ocho semanas, se desarrollen en la sala de informática (o en el aula con el carro de los portátiles) en grupos que permitan ofrecer un equipo informático por alumno.

Las ocho sesiones (pudiendo ser más dependiendo del ritmo de trabajo del alumnado) estarán divididas en una sesión introductoria, dos o tres sesiones de práctica y de afianzamiento, cuatro o cinco sesiones de realización de un proyecto y una última sesión de compartir y valorar las creaciones (tanto en el aula como en la comunidad online).

Las actividades tendrán que estar graduadas teniendo en cuenta la edad del alumnado, ya que ciertas capacidades cognitivas no se adquieren hasta ciertas edades. Por ello nuestra propuesta está dirigida a que al alumnado de primer curso se le pida que realice una tarjeta de felicitación de cumpleaños con letras en movimiento, mientras que al alumnado de sexto curso se le pedirá el uso de una mayor cantidad de comandos para la creación de un programa más complejo, que puede ser un juego básico como el que podemos encontrar en el Anexo I.

El objetivo final de las actividades es la creación de un programa con Scratch y compartirlo con la comunidad online. De esta manera, si el alumnado es capaz de en esas ocho sesiones crear un programa adecuado, podremos comprobar que el objetivo general antes mencionado se ha cumplido. Para cerciorar que el resultado es adecuado, proponemos una serie de criterios de evaluación de las creaciones del alumnado, que podemos ver en la sección siguiente.

4 RESULTADOS Y SU DISCUSIÓN

4.1 Rúbrica de evaluación

Los criterios de evaluación de las creaciones del alumnado de Primaria con Scratch están reflejados en la siguiente tabla, donde también podemos ver el porcentaje de los diferentes aspectos del trabajo realizado.

Tabla 3. Rúbrica de evaluación de las creaciones de Scratch en Primaria[20]

ASPECTOS	%	Excelente	Bien	Regular	Necesita mejorar
		5	4	3	1
Proceso	**20%**	Diligencia correctamente los 5 elementos que componen la plantilla de análisis de problemas: 1) Formular el problema; 2) Resultados esperados; 3) Datos disponibles; 4) Restricciones; 5) procesos necesarios Utiliza productivamente el tiempo asignado para realizar el proyecto. Lo culmina antes del plazo de entrega estipulado. Colabora con sus compañeros, incluso, fuera del tiempo de clase.	Diligencia correctamente 4 de los 5 elementos que componen la plantilla de análisis de problemas: 1) Formular el problema; 2) Resultados esperados; 3) Datos disponibles; 4) Restricciones; 5) procesos necesarios Utiliza productivamente el tiempo asignado para realizar el proyecto. Cumple con el plazo de entrega de este. Colabora adecuadamente con sus compañeros de clase.	Diligencia correctamente 3 de los 5 elementos que comprende la plantilla de análisis de problemas: 1) Formular el problema; 2) Resultados esperados; 3) Datos disponibles; 4) Restricciones; 5) procesos necesarios La mayoría del tiempo de clase lo utiliza para realizar el proyecto. Cumple con dificultad el plazo de entrega. Colabora con sus compañeros de clase en pocas ocasiones.	Diligencia correctamente menos de 3 de los 5 elementos que comprende la plantilla de análisis de problemas: 1) Formular el problema; 2) Resultados esperados; 3) Datos disponibles; 4) Restricciones; 5) procesos necesarios No utiliza productivamente el tiempo asignado para realizar el proyecto. No cumple con el plazo de entrega. No colabora con sus compañeros de clase.
Funcionamiento	**10%**	El programa realizado está completo (cumple con lo planteado por el docente en el proyecto de clase) y funciona correctamente.	El programa realizado no está completo (no cumple con lo planteado por el docente en el proyecto de clase), pero funciona correctamente.	El programa realizado no está completo (no cumple con lo planteado por el docente en el proyecto de clase) y funciona parcialmente.	El programa realizado no está completo (no cumple con lo planteado por el docente en el proyecto de clase) y no funciona.

[20] Información obtenida de la página web (http://www.eduteka.org/modulos/9/280/2173/1, 10/05/2014)

Interfaz gráfica	**10%**	El programa realizado está organizado, tiene varios niveles y su diseño es complejo. La interfaz gráfica es clara, tiene estructura y se adapta tanto al contenido como al diseño del programa. Es fácil interactuar con el programa.	El programa realizado está organizado, tiene dos niveles y su diseño es medianamente complejo. La interfaz gráfica es clara pero tiene poca relación con el contenido y con el diseño del programa. Es fácil interactuar con el programa.	El programa realizado está poco organizado, tiene un solo nivel y su diseño es simple/sencillo. La interfaz gráfica es poco clara y tiene escasa relación tanto con el contenido como con el diseño del programa. Es difícil interactuar con el programa.	El programa realizado no está organizado y su diseño es básico. La interfaz gráfica es confusa. No permite que otras personas puedan interactuar con el programa.
Creatividad	**10%**	El programa realizado es muy original y evidencia un grado de creatividad excepcional por parte del estudiante.	El programa realizado es original y refleja la creatividad del estudiante.	El programa realizado se basa parcialmente en el diseño e ideas de otros. El aporte en creatividad por parte del estudiante es mínimo.	El programa realizado se basa totalmente en el diseño e ideas de otros. No se evidencia ninguna creatividad por parte del estudiante.
Programación	**20%**	El programa evidencia comprensión avanzada de bloques y procedimientos. Utiliza apropiadamente las estructuras de control (secuencial, condicional, iterativa). Los hilos de programación son lógicos y están bien organizados. El programa está correctamente depurado.	El programa demuestra comprensión de los bloques y de cómo estos funcionan en conjunto para alcanzar el resultado esperado. Utiliza apropiadamente algunas estructuras de control (secuencial, condicional, iterativa). Los hilos de programación son lógicos y están organizados. El programa está depurado.	El programa demuestra alguna comprensión de los bloques y cómo éstos funcionan en conjunto. Utiliza deficientemente las estructuras de control (secuencial, condicional, iterativa). Los hilos de programación tienen poca organización. El programa tiene una falla de lógica.	El programa demuestra poca comprensión de los bloques y de cómo éstos funcionan en conjunto. Utiliza equivocadamente las estructuras de control (secuencial, condicional, iterativa). Los hilos de programación carecen de organización. El programa tiene varias fallas de lógica.

Pensamiento Computacional	**20%**	La elaboración del programa evidencia más de 2 características del pensamiento computacional: Recopila datos Analiza datos Representa datos Hace abstracciones Automatiza procesos Simula procesos Ejecuta tareas en paralelo	La elaboración del programa evidencia 2 características del pensamiento computacional: Recopila datos Analiza datos Representa datos Hace abstracciones Automatiza procesos Simula procesos Ejecuta tareas en paralelo	La elaboración del programa evidencia 1 características del pensamiento computacional: Recopila datos Analiza datos Representa datos Hace abstracciones Automatiza procesos Simula procesos Ejecuta tareas en paralelo	La elaboración del programa no evidencia características del pensamiento computacional.
Publicación	**10%**	En la opción "Notas del Proyecto", están completos los datos que identifican el programa en Scratch: nombre del estudiante que lo elaboró, nombre de la Institución Educativa, grado escolar del estudiante, asignatura/materia a la que corresponde el proyecto, y corta descripción del programa.	El programa se encuentra publicado en la cuenta que el estudiante tiene en el sitio Web de Scratch.	El programa se encuentra publicado en una cuenta cualquiera del sitio Web de Scratch.	El programa NO está publicado en el sitio Web de Scratch. En la opción "Notas del Proyecto", NO están completos los datos que identifican el programa en Scratch.
	100 %	**NOTA FINAL:**		**___**	

4.2 Discusión de los resultados

Como hemos podido ver en la rúbrica, la evaluación de las creaciones requiere analizar y observar tanto el trabajo final del alumnado como el proceso de creación.

Por otra parte, es lógico pensar que una parte del alumnado con necesidades educativas específicas deberá de ser evaluado de manera más personalizada, por lo que se deberían realizar las adaptaciones pertinentes en los criterios de evaluación.

Para que nuestro objetivo de comprobar si el uso de Scratch en Educación Primaria se cumpla, y tras comprobar que los resultados son fiables y adecuados a la realidad, tendremos que decidir qué porcentaje de trabajos aprobados es el mínimo necesario.

En caso de que los resultados de la evaluación sean notablemente negativos, con más de la mitad del alumnado suspendido, significará que nuestra propuesta no se adapta a la realidad del aula, por lo que habría que replantearla.

En caso de que los resultados sean medianamente aceptables, con la mayoría del alumnado aprobado aunque pocos pasen del 85%, significará que la propuesta es viable, ya que supone un reto cumplible para el alumnado, pero a su vez queda margen para el aprendizaje más profundo.

En caso de que los resultados sean mayoritariamente excelentes, significará que nuestra propuesta es viable, aunque habría que añadirle un mayor grado de complejidad para que suponga un reto y un aprendizaje para el alumnado. El programa Scratch, gracias a sus ilimitadas posibilidades, permite ese aumento de complejidad.

CONCLUSIONES Y CUESTIONES ABIERTAS

A modo de conclusión, habiendo visto la historia, evolución y estado actual de la programación informática, e intuyendo un futuro en expansión, podemos decir que es un acierto el incluir contenidos de programación en Educación Primaria en Navarra, utilizando un programa adaptado como es Scratch.

A través de la realización de este trabajo nos hemos convencido de que Scratch es un material adecuado para utilizarlo en las aulas de Primaria, al igual que diferentes estudios y autores afirman. Aunque para tener la certeza de ello tendríamos que observar los resultados del estudio de caso propuesto.

De todos modos, Scratch abre un amplio abanico de posibilidades para los docentes, y permite el trabajo complementario en las diferentes áreas del currículo. Además, es compatible con métodos actuales como el trabajo por proyectos, y como hemos podido ver también podemos relacionarlo con las leyes educativas.

Sin embargo, para poder introducir Scratch a nivel de Navarra, se nos plantean ciertas cuestiones como la cantidad de equipos informáticos en las escuelas y una posible inversión dotacional o la preparación de cursos de formación para docentes.

Quedan también abiertas ciertas cuestiones más generales como si habría que incluir explícitamente la programación informática y el pensamiento computacional en el currículo o si la programación se entenderá en el futuro como una habilidad fundamental como son el leer y el escribir.

REFERENCIAS

Badger, M. (2009). *Scratch 1.4: Beginner's Guide*. Packt Publishing Ltd.

Balagurusamy, E. (2010). *Fundamentals of Computers*. Tata Mcgraw Hill Education Pvt. Limited.

Bruckman, A. & Resnick, M. (1995). The MediaMoo project: constructionism and professional community. *Convergence: The International Journal of Research into New Media Technologies*, *1*(1), 94-109.

Feurzeig, W., Papert, S. A. & Lawler, B. (2011). Programming-languages as a conceptual framework for teaching mathematics. *Interactive Learning Environments*, *19*(5), 487-501. doi: 10.1080/10494820903520040

Hartle, R. T., Baviskar, S. & Smith, R. (2012). A Field Guide to Constructivism in the College Science Classroom: Four Essential Criteria and a Guide to Their Usage. *Bioscene: Journal of College Biology Teaching*, *38*(2), 31-35.

López García, J. C. (2009). Algoritmos y Programación: Guía para docentes. Recuperado de http://www.eduteka.org/pdfdir/AlgoritmosProgramacion.pdf

Pea, R. D. & Kurland, D. M. (1984). On the cognitive effects of learning computer programming. *New ideas in psychology*, *2*(2), 137-168.

Resta, P. & Laferrière, T. (2007). Technology in support of collaborative learning. *Educational Psychology Review*, *19*(1), 65-83.

Wilson, A. & Moffat, D. C. (2010). Evaluating Scratch to introduce younger schoolchildren to programming. *Proceedings of the 22nd Annual Psychology of Programming Interest Group (Universidad Carlos III de Madrid, Leganés, Spain*.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *366*(1881), 3717-3725.

ANEXOS

Anexo I

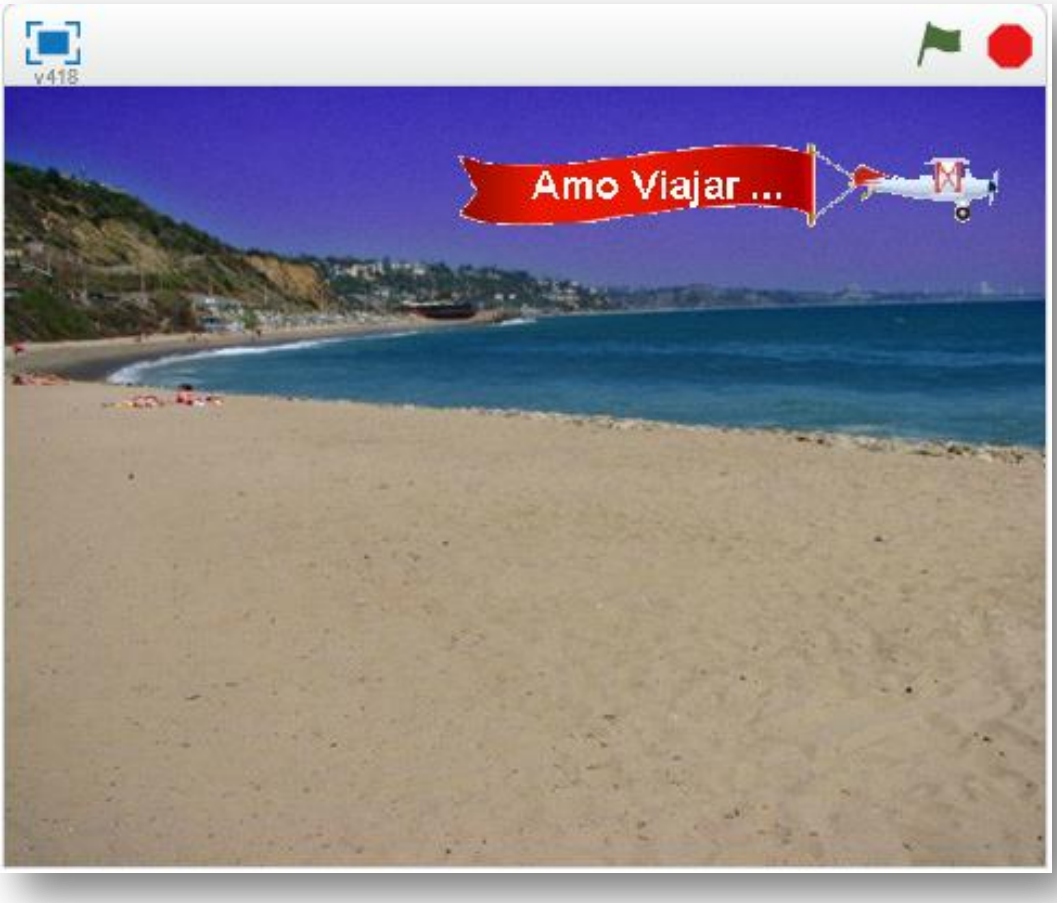
Diez actividades cortas para aprender Scratch[21]:

- Mañana, tarde, noche: con esta actividad, según el objeto que se presione (botón amanecer, sol, luna), debe lograrse que cambie el escenario y despliegue tres fondos diferentes, “amanecer”, “mediodía” y “noche”. Objetivo: desarrollar o afianzar la habilidad en el uso de los comandos "enviar a todos" y "al recibir".

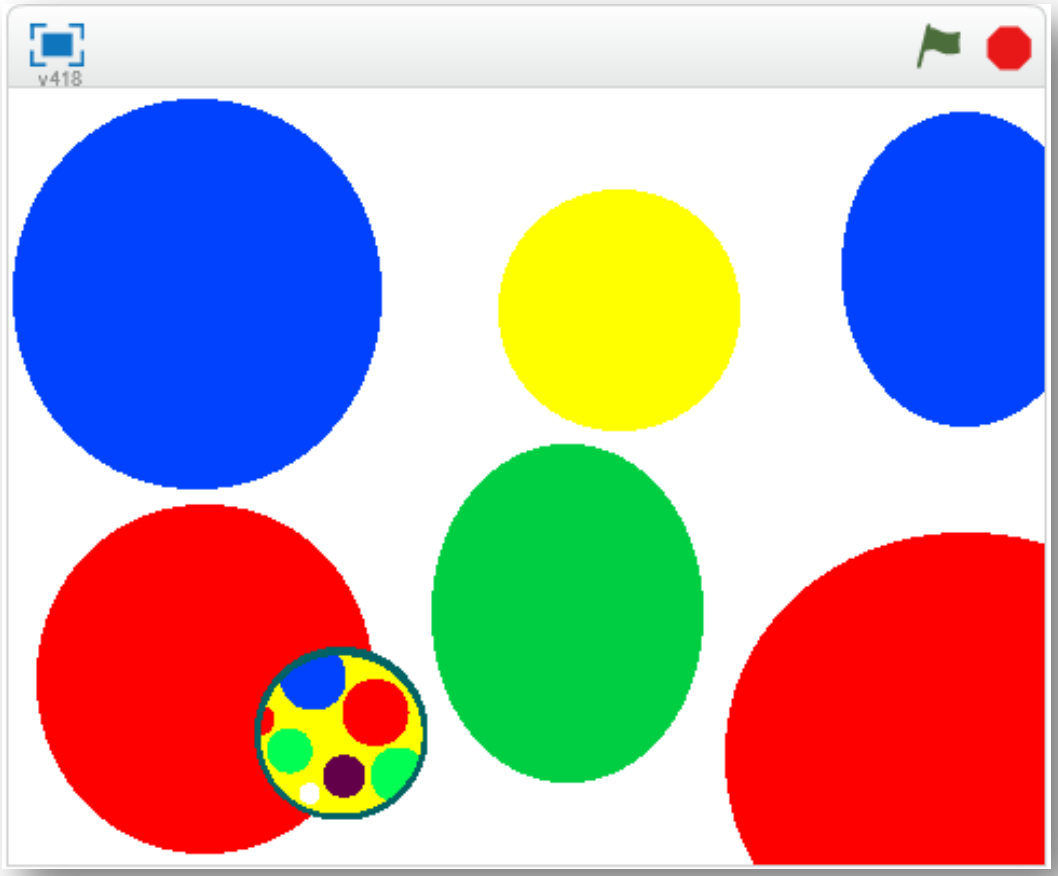
[21] Información obtenida de la página web (http://www.eduteka.org/modulos/9/281/2131/1, 07/05/2014)

- Globos que giran: dados tres objetos, cuando se presione uno de ellos los otros dos deben girar. Objetivo: desarrollar o afianzar la habilidad en el manejo de los comandos "enviar a todos" y "al recibir".

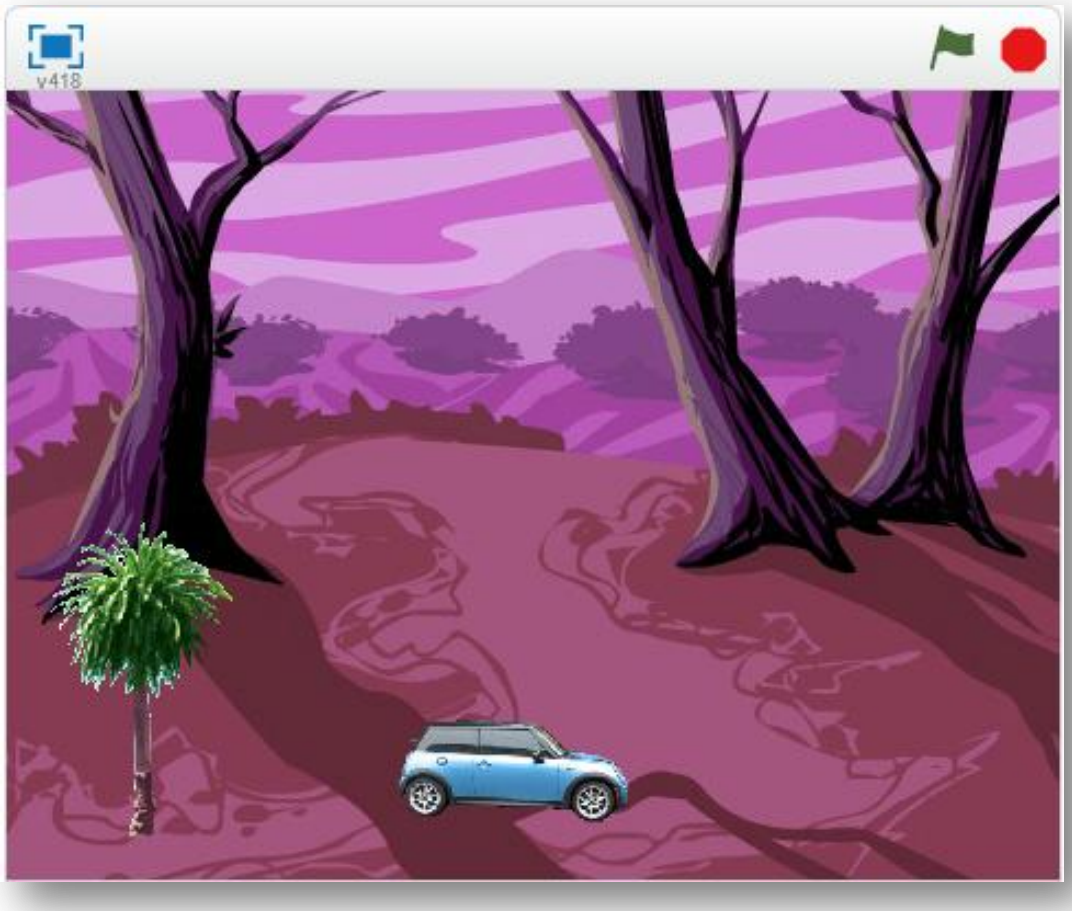
- Avioneta con banner (bandera) y paisaje: una avioneta se desplaza de izquierda a derecha por el escenario, cuando toca el borde derecho del escenario este debe cambiar a otro fondo y la avioneta debe iniciar de nuevo su desplazamiento desde el borde izquierdo. Debe contarse al menos con 3 fondos y la avioneta continua moviéndose por siempre a través de ellos. Objetivo: desarrollar o afianzar habilidad en el manejo de los comandos "enviar a todos" y "al recibir".



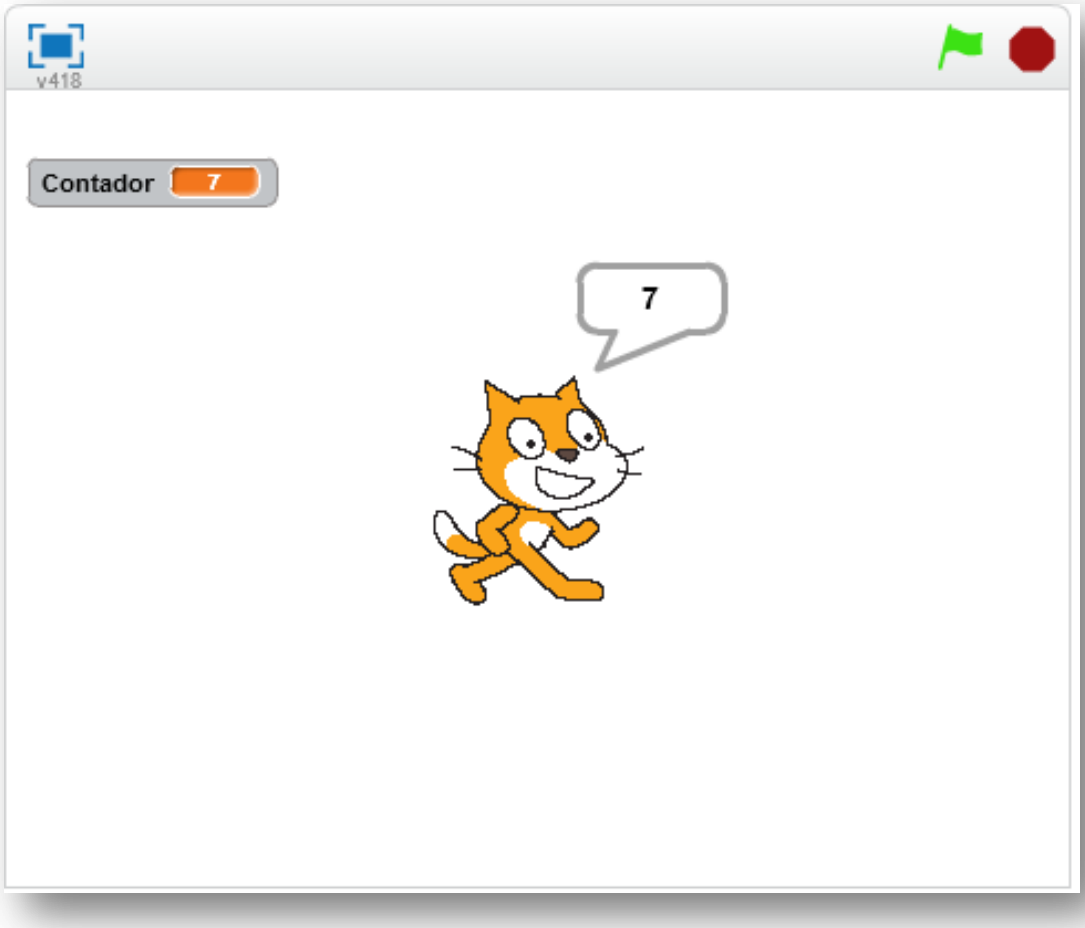
- Balón y sonido: un objeto (“balón”) se desplaza por un fondo en el que algunas partes tienen diferentes colores. Cuando el objeto toca un color debe reproducir algún sonido (un sonido diferente por color). Objetivo: desarrollar o afianzar habilidades en el manejo de eventos y condicionales.



- Mostrar y esconder objetos: dados tres objetos (“palmera, “coche” y “caballo”), asociar cada uno a un contexto diferente (“playa”, “ciudad” o “campo”). Al presionar un objeto, este se debe esconder y el fondo debe cambiar al contexto que le corresponde; los otros dos objetos deben permanecer visibles. Siempre deben permanecer en el escenario los dos objetos a los cuales no les corresponde el fondo visible. Objetivo: desarrollar o afianzar la habilidad en los comandos "enviar a todos", “al recibir”, “mostrar” y “esconder” objeto.

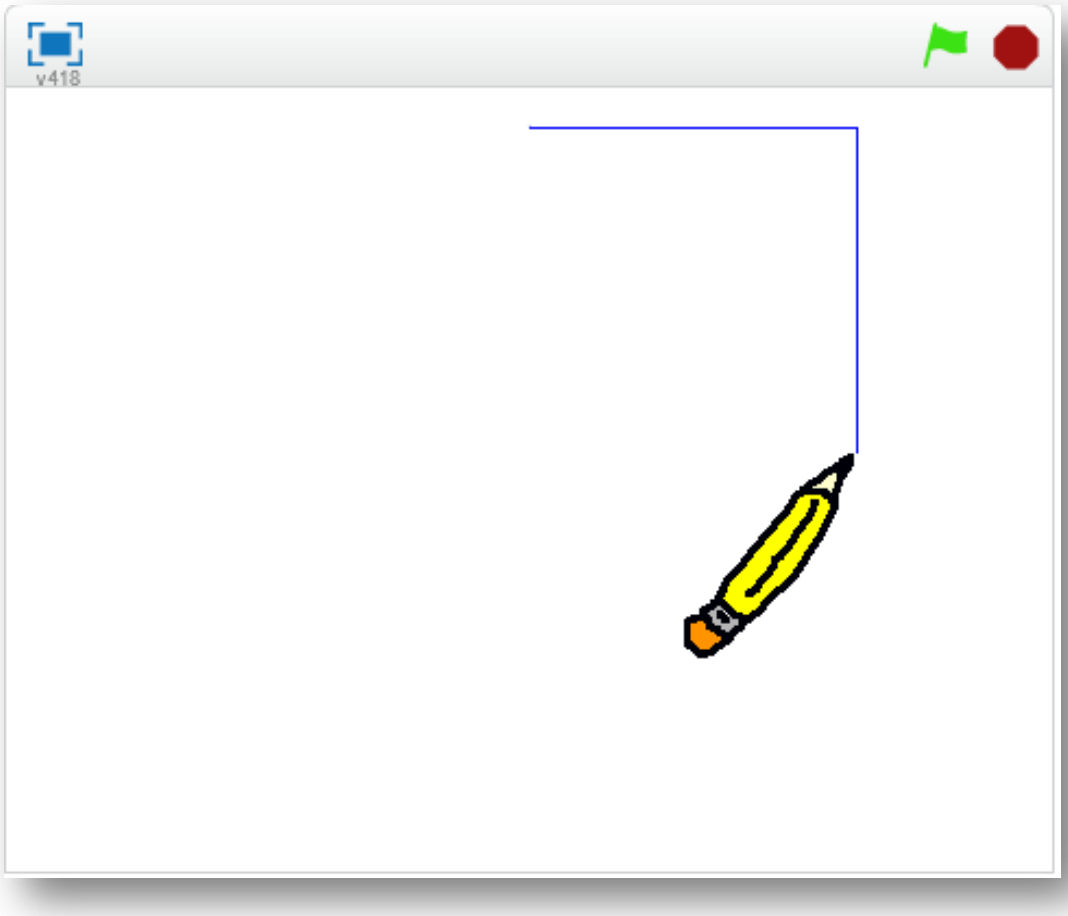


- Conteo: en este proyecto se solicita generar un contador de los números del 1 al 10. Los números se deben mostrar en una variable desplegada en pantalla. Objetivo: practicar el uso de variables.

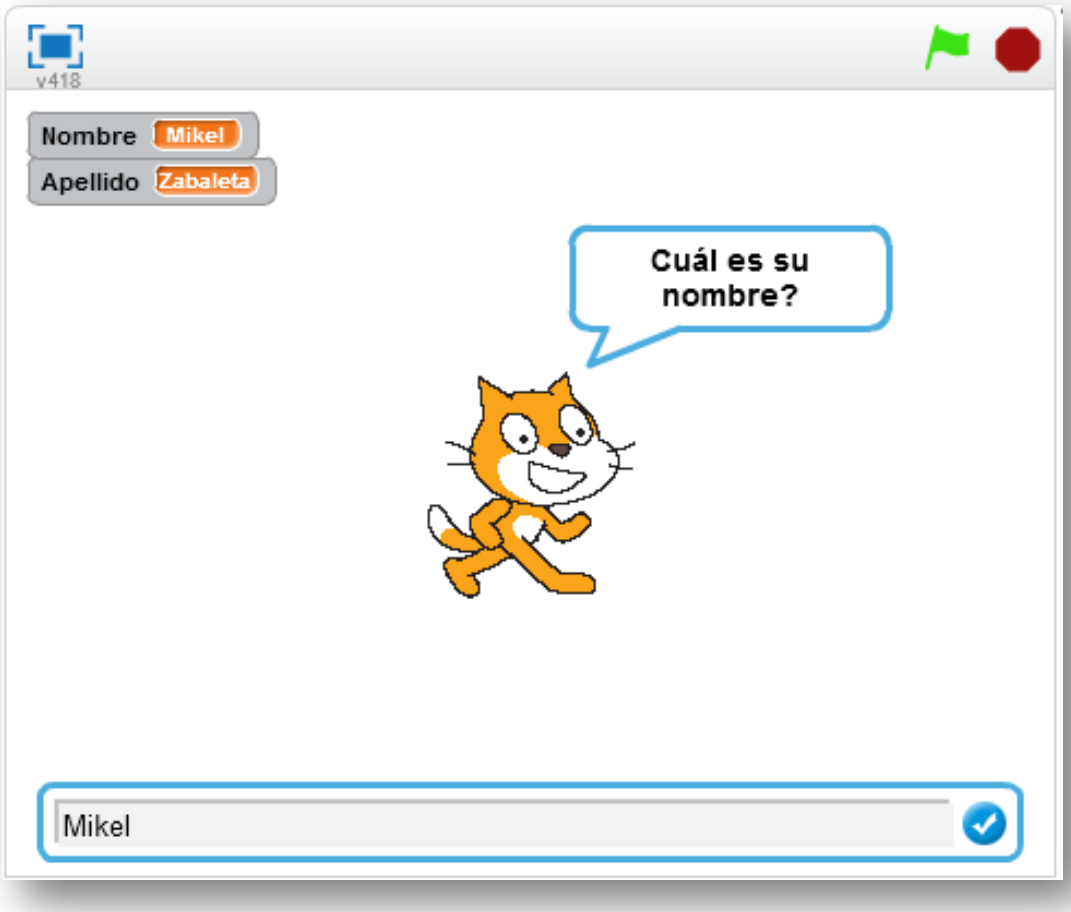


- Sumar dos números: se deben solicitar dos números por pantalla, sumarlos y mostrar el resultado. Se deben tener las variables desplegadas en el escenario. Objetivo: Practicar el manejo de variables y el comando preguntar.

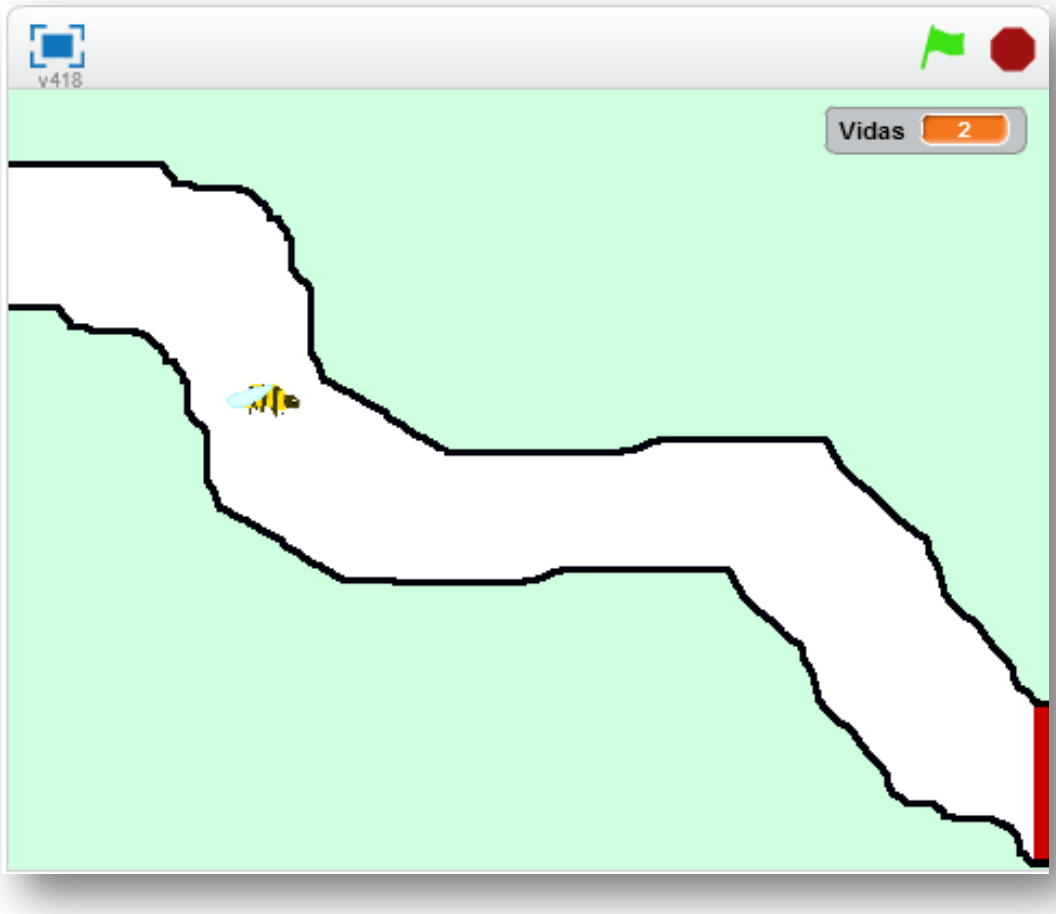
- Dibujo básico de figuras geométricas: como este es un ejercicio de ejemplo con tres figuras, sin menú de opciones, cada bloque está despegado del bloque "al presionar bandera verde". Cada vez que se vaya a probar una figura se debe pegar el bloque correspondiente y despegar los de las otras dos figuras. Objetivo: practicar el uso de los comandos correspondientes a la categoría “Lápiz”.



- Nombre y apellido: pedir por pantalla el nombre y el apellido. Luego utilizar los comandos “unir” y "decir" para mostrar el nombre completo (nombre seguido de apellido). Objetivo: practicar los comandos preguntar, unir, decir y fijar variable.



- Juego básico (laberinto): en este juego, deben utilizarse las flechas de desplazamiento que tiene el teclado para mover la abeja a lo largo de la franja blanca. Si la abeja toca la línea negra del camino, debe regresar a la posición inicial y disminuir una vida. Cuando se agoten todas las vidas, debe aparecer un fondo con un letrero que diga “PERDIÓ”. Si la abeja avanza y toca la línea roja de meta, debe aparecer un fondo con un letrero que diga “GANÓ”. Objetivo: integrar los comandos utilizados en las actividades 1 a 9.

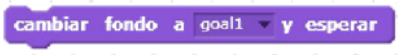


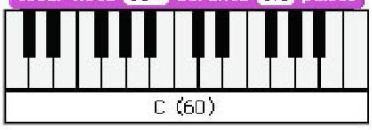
Anexo II

Descripción de los bloques de Scratch (por categorías)[22]:

MOVIMIENTO	
	Mueve el Objeto hacia adelante o hacia atrás.
	Rota el Objeto en el sentido de las manecillas del reloj.
	Rota el Objeto en el sentido contrario a las manecillas del reloj.
	Apunta el Objeto en la dirección especificada (0=arriba; 90=derecha; 180=abajo; -90=izquierda).
	Apunta el Objeto hacia el puntero del ratón o hacia otro Objeto.
	Mueve el Objeto hacia una posición específica de X, Y en el escenario.
	Mueve el Objeto a la ubicación del puntero del ratón o de otro Objeto.
	Mueve el Objeto suavemente a una posición determinada en un lapso de tiempo específico.
	Cambia la posición X del Objeto en una cantidad determinada (incrementa).
	Fija la posición X del Objeto a un valor específico.
	Modifica la posición Y del Objeto en una cantidad determinada (incrementa).
	Fija la posición Y del Objeto a un valor específico.
	Gira el Objeto en sentido contrario, cuando este toca un borde del escenario.
	Fija el estilo de Rotación así: En todas direcciones (el disfraz rota a medida que el Objeto cambia de dirección); Izquierda-derecha (el disfraz mira o a la izquierda o a la derecha); No rotar (el disfraz nunca rota, aun cuando el Objeto cambie de dirección).
	Informa la posición X del Objeto. (Rango entre –240 a 240)
	Informa la posición Y del Objeto (Rango entre -180 a 180)
	Informa la dirección del Objeto (0=arriba; 90=derecha; -90=izquierda; 180=abajo)

[22] Información obtenida de la página web (http://www.eduteka.org/pdfdir/ScratchGuiaReferencia.pdf, 09/05/2014)

APARIENCIA	
	Despliega una nube de diálogo del Objeto durante un lapso de tiempo determinado.
	Despliega una nube de diálogo del Objeto (se puede eliminar esta burbuja de diálogo ejecutando este bloque sin texto alguno).
	Despliega una nube de pensamiento del Objeto durante un determinado lapso de tiempo.
	Despliega una nube de pensamiento del Objeto.
	Hace aparecer un Objeto en el escenario.
	Hace desaparecer un Objeto del escenario (cuando el Objeto está escondido, otros Objetos no lo pueden detectar con el bloque "¿tocando?").
	Modifica la apariencia del Objeto cambiando de disfraz.
	Cambia el disfraz del Objeto por el siguiente disfraz en la lista de disfraces (cuando llega al final del listado de estos, vuelve a comenzar con el primer disfraz).
	Modifica la apariencia del escenario pasando al siguiente fondo disponible en el listado de estos.
	Modifica la apariencia del escenario cambiando a un fondo diferente.
	Cambia el escenario a un fondo específico, al siguiente fondo o al fondo anterior.
	Modifica (incrementa o reduce) un efecto visual del Objeto en una cantidad especificada (use el menú desplegable para seleccionar el efecto).
	Establece un efecto visual a un número dado (la mayoría de efectos visuales se ubica en un rango de 0 a 100).
	Limpia o borra todos los efectos gráficos de un Objeto
	Modifica el tamaño del Objeto en una cantidad especificada (incrementa o reduce).
	Ajusta el tamaño del Objeto en un porcentaje (%) específico respecto a su tamaño original.
	Ubica el Objeto al frente de todos los demás Objetos (capa superior).
	Mueve el Objeto hacia atrás, un número determinado de capas, de manera que pueda ocultarse detrás de otros Objetos.
	Reporta el número del fondo actual del escenario.
	Informa el número correspondiente al disfraz actual del Objeto.
	Informa en nombre del fondo actual.
	Informa el tamaño del Objeto como porcentaje (%) de su tamaño original.

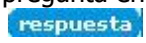
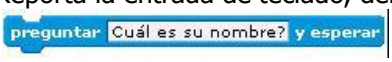
SONIDO	
tocar sonido miau	Comienza la reproducción del sonido seleccionado del menú desplegable, e inmediatamente pasa al siguiente bloque aunque el sonido se esté ejecutando aún.
tocar sonido miau y esperar	Reproduce un sonido y espera hasta que el sonido termine, antes de continuar con el bloque siguiente.
detener todos los sonidos	Detiene todos los sonidos.
tocar tambor 48 durante 0.2 pulsos	Reproduce un determinado número de sonido de tambor, seleccionado del menú desplegable, durante un número específico de pulsos.
silencio por 0.2 pulsos	Descansa, no toca nada, durante un número específico de pulsos.
tocar nota 60 durante 0.5 pulsos 	Reproduce una nota musical (número altos para tonos altos) durante un número específico de pulsos.
fijar instrumento a 1	Establece el tipo de instrumento que usa el Objeto para los bloques de "tocar notas" (cada Objeto tiene su propio instrumento).
cambiar volumen por -10	Modifica el volumen del sonido del Objeto en un valor especificado (incrementa o reduce el volumen).
fijar volumen a 100	Fija el volumen del sonido del Objeto a un valor especifico.
volumen	Informa el volumen del sonido del Objeto.
cambiar tempo por 20 fijar tempo a 60 ppm	Modifica el *tempo* del Objeto en una cantidad específica (incrementa o reduce). Fija el tempo del Objeto a un valor especificado de pulsos por minuto.
tempo	Informa el tempo del Objeto en pulsos por minuto.
LÁPIZ	
borrar	Borra todas las marcas de lápiz y de sellos (estampados) del Escenario.
sellar	Estampa o copia la imagen del Objeto en el Escenario.
bajar lápiz	Baja el lápiz del Objeto, de manera que este pinte a medida que se mueve.
subir lápiz	Levanta el lápiz del Objeto, de manera que no pinte cuando se mueva.
fijar color de lápiz a	Establece el color del lápiz, basado en la selección hecha en la paleta de color.
cambiar color del lápiz por 10	Modifica el color del lápiz en una cantidad específica (incrementa o reduce).
fijar color de lápiz a 0	Establece el color del lápiz a un valor determinado. (color-lápiz=0 en el borde rojo del arco iris; color-lápiz=100 en el borde azul del arco iris. Rango de 0 a 200 para ir a través de la paleta de colores)
cambiar intensidad de lápiz por 10	Modifica la intensidad del lápiz en una cantidad especificada (incrementa o reduce).
fijar intensidad de lápiz a 50	Establece un valor específico para la intensidad del lápiz (sombra-lápiz=0 es muy oscura; sombra-lápiz=100 es muy clara. El valor por defecto es 50, a menos que se establezca con la paleta de color).
cambiar tamaño de lápiz por 1	Cambia el grosor del lápiz en una cantidad específica (incrementa o reduce en una cantidad específica).
fijar tamaño de lápiz a 1	Establece el grosor del lápiz.

DATOS	
	Permite crear y nombrar una nueva variable. Cuando usted crea una variable, aparecen los bloques correspondientes a ella. Se puede escoger si la variable es para todos los Objetos (global) o solo para un Objeto (local). También permite definir si los datos de la variable se almacenarán en el servidor Scratch del MIT.
	Informa el valor de la variable
	Fija la variable a un valor específico.
	Modifica (incrementa o reduce) la variable en una cantidad determinada (Si se tiene más de una variable, utilice el menú desplegable para seleccionar el nombre de la variable)
	Muestra el monitor de la variable en el escenario
	Esconde el monitor de la variable para que no aparezca en el escenario
	Permite crear y nombrar una nueva lista. Cuando se genera una lista,

	aparecen los bloques para esa lista. Se puede escoger si la lista es para todos los Objetos (global) o solo para un Objeto (local)
	Reporta (muestra) todos los elementos que tiene la lista.
	Adiciona el elemento especificado al final de la lista (el elemento puede ser un número o una cadena de letras u otros caracteres).
	Borra uno o todos los elementos de una lista. Se puede escoger del menú desplegable o usar un número para indicar qué elemento borrar. Si escoge "último" borrará el último elemento de la lista. Si escoge "todos" borrará todo lo que contiene la lista. Borrar, reduce la longitud de la lista.
	Inserta un elemento en un lugar específico de la lista. Se puede escoger del menú desplegable o usar un número para indicar dónde insertar el elemento dentro de la lista. Si escoge "último" adiciona el elemento al final de la lista. Si se escoge "cualquiera" lo inserta aleatoriamente en la lista. La longitud de la lista se incrementa en 1.
	Reemplaza un elemento de la lista con un valor específico. Se puede escoger del menú desplegable o usar un número para especificar el elemento que va a reemplazar. Si escoge "último" , reemplaza el último elemento de la lista. Si escoge "cualquiera" reemplaza aleatoriamente un elemento de la lista. La longitud de la lista no se modifica.
	Reporta el elemento en una ubicación específica dentro de la lista. Usted puede especificar cuál elemento, eligiendo del menú desplegable o escribiendo un número.
	Reporta cuántos elementos hay en la lista.
	Informa verdadero si la lista contiene el elemento especificado. El ítem debe coincidir perfectamente para reportarse como verdadero.
	Muestra en el escenario los elementos que tiene la lista.
	Esconde del escenario los elementos de una lista.

EVENTOS	
	Ejecuta el programa que tiene debajo al hacer clic en la bandera verde.
	Ejecuta el programa que tiene debajo al presionar una tecla específica.
	Ejecuta el programa que tiene debajo al hacer clic en un Objeto.
	Ejecuta las instrucciones que tiene debajo cuando el fondo cambie al que se determina en esta instrucción.
	Ejecuta las instrucciones que tiene debajo cuando la intensidad del sonido sea mayor a lo estipulado en la instrucción.
	Ejecuta el programa que tiene debajo cuando recibe un mensaje específico "enviar a todos".
	Envía un mensaje a todos los Objetos y luego continúa con el bloque siguiente sin esperar a que se realicen las acciones de los Objetos activados.

CONTROL	
	Espera un número determinado de segundos y continúa luego con el bloque siguiente.
	Ejecuta, un número específico de veces, los bloques en su interior.
	Ejecuta continuamente los bloques en su interior.
	Si la condición es verdadera, ejecuta los bloques en su interior.
	Comprueba continuamente si una condición es verdadera; cada que es verdadera, ejecuta los bloques en su interior.
	Si la condición es verdadera, ejecuta los bloques dentro de la porción "si"; si no, ejecuta los bloques que están dentro de la porción "si no".
	Espera hasta que la condición sea verdadera, para ejecutar los bloques siguientes.
	Comprueba si la condición es falsa; si lo es, ejecuta los bloques en su interior y vuelve a chequear la condición. Si la condición es verdadera, pasa a los bloques siguientes.
	Detiene el programa (que se está ejecutando dentro de un Objeto). Detiene todos los programas de todos los Objetos.
	Le dice a un clon qué hacer una vez éste se ha creado.
	Crea clones de un objeto determinado. El clon es un duplicado que sólo existe mientras el proyecto está ejecutándose.
	Borra un objeto clonado.

SENSORES	
	Informa verdadero, si el Objeto está tocando un Objeto específico, un borde o el puntero del ratón (seleccionados del menú desplegable).
	Informa verdadero, si el Objeto está tocando un color específico. (Haga clic en la paleta de color y luego utilice el gotero para seleccionar el color).
	Reporta verdadero si el primer color (dentro del Objeto), está tocando un segundo color (tanto en el fondo como en otro Objeto). Haga clic en la paleta de color y luego utilice el gotero para seleccionar el color.
	Informa la distancia desde un Objeto específico o desde el puntero del ratón.
	Formula una pregunta en la pantalla y guarda lo que se ingresa por teclado en la  . Hace que el programa espere hasta que se presione la tecla "Enter" o se haga clic en la casilla de verificación.
	Reporta la entrada de teclado, del uso más reciente de  . Se comparte para todos los Objetos (Global)
	Informa verdadero, si una tecla específica está presionada.
	Informa verdadero, si el botón del ratón está presionado.
	Informa la posición "X" del puntero del ratón.
	Informa la posición "Y" del puntero del ratón.
	Reporta el volumen de los sonidos captados por el micrófono del computador (entre 1 y 100).
	Detecta la cantidad de movimiento actual en la imagen de vídeo. También puede detectar la dirección del movimiento.
	Bloque para encender, apagar o invertir el video.
	Fija en un porcentaje la transparencia del video.
	Reporta el valor del cronometro en segundos (el cronómetro siempre está contando).
	Fija el cronómetro en 0.
	Informa una propiedad o variable de otro Objeto.
	Reporta varios valores del sistema: año, mes, fecha, día de la semana, hora, minuto, segundo.
	Reporta el número de días transcurridos desde enero 1 del 2000.
	Reporta el nombre del usuario.

OPERADORES	
	Suma dos números.
	Resta dos números (Sustrae el segundo número de el primero)
	Multiplica dos números.
	Divide dos números (Divide el primer número entre el segundo)
	Selecciona al azar un número entero dentro de un rango especificado.
	Informa verdadero, si el primer valor es menor que el segundo.
	Reporta verdadero, si dos valores son iguales.
	Informa verdadero, si el primer valor es mayor que el segundo.
	Informa verdadero, si ambas condiciones son verdaderas.
	Informa verdadero, si una de las dos condiciones es verdadera.
	Reporta verdadero, si la condición es falsa; reporta falso si la condición es verdadera.
	Concatena (combina) cadenas de letras (caracteres)
	Informa la letra en una posición específica dentro de una cadena
	Informa el número de letras en una cadena
	Informa el residuo (módulo) de la división del primer número entre el segundo número.
	Informa el entero más cercano a un número.

MÁS BLOQUES	
	Permite asignar un nombre a un bloque propio y luego programarle lo que debe hacer. Estos bloques también se llaman procedimientos. Los nuevos bloques se crean solo para el objeto en el que se esté ubicado.
	Esta instrucción se crea automáticamente en el área de programas. Debajo se ubican los bloques que constituyen el programa del bloque.
	Bloque nuevo que se puede utilizar en cualquier programa del objeto en el cual fue creado.